Downloaded from orbit.dtu.dk on: May 04, 2023

DTU Library

Optimisation in liner shipping: Challenges in mega vessel's cargo-mix and loading operations

Christensen, Jonas Mark

Publication date:
2017

Document Version
Publisher's PDF, also known as Version of record

Link back to DTU Orbit

Citation (APA):
Christensen, J. M. (2017). *Optimisation in liner shipping: Challenges in mega vessel's cargo-mix and loading operations*.

General rights
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain
- You may freely distribute the URL identifying the publication in the public portal

If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.

Optimisation in liner shipping:
Challenges in mega vessel's cargo-mix and loading operations

Jonas Mark Christensen

Kongens Lyngby 2017

Technical University of Denmark
DTU Management Engineering
Management Science
Produktionstorvet, building 424
2800 Kongens Lyngby, Denmark
Phone +45 4525 4800
www.man.dtu.dk

Summary (English)

The goal of this thesis is to develop decision support tools, to optimise the planning of certain activities within the liner shipping industry. With the emergence of mega-vessels, the planning of vessel and terminal activities are becoming more and more vital for the liner shipping industry. This thesis considers two specific problems where optimisation methods can be used to help with the planning. The first of these problems aim to improve the utilisation of the vessels, and the second aim to improve terminal productivity to minimise the time spent at ports for vessels.

For large vessels to achieve economies of scale, it is crucial they be fully utilised. E.g. It requires a 91% utilisation for an 18000 TEU vessel to be cost-effective over a fully loaded 14000 TEU vessel. When determining a load configuration (stowage plan), it is of utmost importance to ensure the vessel do not capsize or break. For this, the weight distribution of the containers on the vessel plays an essential role. A stowage plan must also conform with the cargo already loaded on the vessel, the cargo to be loaded while also ensuring the vessel can still be utilised to its maximum in downstream ports. Given the growth in the size of newly built vessels, this task is becoming harder and harder, as well as more important. We study this problem and develop an adaptable framework to help with different kinds of what-if analysis.

While the voyage cost per container is decreased for the mega-vessels, the handling cost is increased. Hence, terminals are under pressure to increase productivity and minimise the turnaround time for the vessels. The design of the mega-vessels, however, makes this hard. These vessels are both higher and wider, requiring cranes to reach further away from the quay and deeper into the vessel. As an effect of the inherent design of the vessels, the (un)loading time per container is thus increased, compared to smaller vessels. With more mega-vessels coming shortly it is also expected that yard congestion will be an increasing problem. Without any significant improvements it is therefore not expected that productivity will increase, and even maintaining the current level will be a challenge. Decreasing the time spent at port is a shared goal between the liner and the terminal. For the liner, the operating costs are decreased, while the terminal can plan the use of their container-handling equipment better, so it can be used for another vessel. We investigate such a collaboration and how the sharing of data can be used to optimise the terminal-side planning while ensuring the liner also benefits from this.

The results are prototypes of decision support tools, that can be used to automatize the planning of these activities.

Resumé (Summary in Danish)

Målet for denne afhandling er at udvikle beslutningsstøtteværktøjer, der kan bruges til at optimere planlægningen af aktiviteter inden for container shipping. Med introduktionen af mega-skibe bliver planlægningen af skibs- og terminalaktiviteter mere og mere afgørende for industrien. Denne afhandling omhandler to specifikke problemer, hvor optimeringsmetoder kan bruges til at hjælpe med planlægningen. Det første af disse problemer sigter mod at forbedre udnyttelsen af skibene, og det andet fokuserer på at forbedre terminalproduktiviteten for at minimere tiden i havn for skibene.

For at store skibe kan opnå stordriftsfordele er det afgørende, at kapaciteten udnyttes til fulde. Det kræver for eksempel mindst en udnyttelsesgrad på 91% for at et 18000 TEU-skib er omkostningseffektivt sammenlignet med et fuldt lastet 14000 TEU-skib. Ved bestemmelse af en belastningskonfiguration (stuvnings plan) er det yderst vigtigt at sikre, at skibet ikke kæntrer eller knækker. Her spiller vægtfordelingen af containerne på skibet en afgørende rolle. En stuvnings plan skal passe med containerne der allerede er lastet på skibet, containerne der skal lastes, samtidig skal det også sikres at skibet stadig kan udnyttes til fulde i de kommende havne. I forbindelse med at nybyggede skibe bliver større og større bliver det sværere at bestemme en god stuvning plan, samtidig med at det også bliver vigtigere. Vi betragter dette problem og udvikler en fleksibel matematisk metode til at hjælpe med forskellige former for what-if analyser.

Mens fragt omkostningerne pr. container er faldet for mega-skibene, er håndteringsomkostningerne steget. Derfor er terminalerne under pres for at øge produktiviteten og minimere ekspeditionstiden for skibene. Designet af mega-skibene gør det imidlertid svært. Skibene er både højere og bredere, hvilket kræver at kranerne skal række længere væk fra kajen og dybere ned i skibet. Som en effekt af skibenes design øges (af)lastetiden per container i forhold til mindre skibe. Med flere mega-skibe på vej i den nærmeste fremtid, forventes det også at trafikpropper i containerterminalen vil være et stigende problem. Uden væsentlige forbedringer forventes det derfor ikke, at produktiviteten vil stige, og selv at opretholde det nuværende niveau vil være en udfordring. At reducere tiden i havnen er et fælles mål mellem rederiet og terminalen. Driftsomkostningerne for skibene falder, samtidig med at terminalen kan planlægge brugen af deres containerhåndteringsudstyr bedre, så det kan bruges til andre skibe. Vi undersøger et sådant samarbejde og hvordan dataudveksling kan bruges til at optimere terminalplanlægningen, samtidig med at skibene også drager fordel af dette.

Resultaterne er prototyper af beslutningsstøtteværktøjer, der kan bruges til at automatisere planlægningen af disse aktiviteter.

Preface

This thesis was carried out at the Division of Management Science, DTU Management Engineering, Technical University of Denmark during the period September 2014 - October 2017. The thesis constitutes a partial fulfilment of the requirements for acquiring a Ph.D. in Engineering. Associate Professor Dario Pacino supervised the project, and Professor Harilaos Psaraftis acted as co-supervisor.

The thesis deals with challenges in the liner shipping industry due to increasing vessel sizes. Two specific problems are detailed, and solution methods are proposed to mitigate these problems. The solution methods are based on Operations Research techniques, and it is expected that the reader has knowledge of such methods.

The thesis consists of three main parts. First is the introduction which gives a thorough presentation of the liner shipping industry, and the covered problems. The next part contains three research papers written as part of the Ph.D. The last part contains additional work related to the research papers. This work is currently unpublished, but we believe it can be the basis of one or more journal papers.

This work has been funded by the Innovation Fund Denmark (IFD) under the GREENSHIP Project (project number: 1313-00005B-GREENSHIP). IFD has supported this work solely financially and has not taken part in any research-related activities.

Lyngby, 26-October-2017

Jonas Mark Christensen

Acknowledgements

Many people have helped me on the road to finish this Ph.D. I would like to thank Professor David Pisinger and Professor Stefan Røpke for both being excellent teachers and supervisors during my bachelors and master studies. Without you, I would not have found the beauty of OR or considered doing a Ph.D. David, you are an excellent and engaging teacher. I had the pleasure to have you as my teacher on my first OR course, and after that course, I knew I wanted to do more within the field. Stefan, I would like to thank you for being an enthusiastic supervisor on multiple projects during my master studies.

The biggest thanks of them all must go to my supervisor team: Associate Professor Dario Pacino and Professor Harilaos Psaraftis. Dario, I am sincerely grateful that you saw the potential and encouraged me to do a Ph.D. You have supported me all the way to become a better researcher and deliver high-quality research, and I would like to thank you for all your constructive comments and always having an open door.

Special thanks also go to my colleagues at DTU Management Engineering, and especially my officemates, João Fonseca, George Panagakos and Çağatay Iris. I have not only enjoyed sharing an office with you but at times also shared my frustrations.

I would also like to thank Professor Alan Erera for giving me the opportunity to visit Georgia Institute of Technology and the staff that helped me make it possible. I truly enjoyed my time in Georgia, and these memories will always stay with me.

I am grateful to my father and mother for their love and support. I have always enjoyed our family gatherings in Frederikssund, and they have always given me the energy to work towards the next goal. To my girlfriend Molly: I am grateful to you for giving me space and let me focus on finishing this thesis. I am looking forward to giving you the attention you deserve.

Last but not least, I am thankful to DTU for awarding me the Ph.D. scholarship and to Innovation Fund Denmark who has funded the project. Likewise, I would like to thank, Knud Højgaards Fond, Oticon Fonden, P.A. Fiskers fond, Danmark-Amerika Fondet, Den Danske Maritime Fond, Kaj og Hermilla Ostenfelds Fond for helping make my external research stay possible.

Contents

Summary (English) i

Resumé (Summary in Danish) iii

Preface v

Acknowledgements vii

1 Introduction 1
- 1.1 Thesis outline . . . 4
- 1.2 Introduction to maritime logistics . . . 5
  - 1.2.1 Industrial and tramp shipping . . . 5
  - 1.2.2 Liner shipping . . . 5
- 1.3 Liner shipping and operations research . . . 12
  - 1.3.1 Network design . . . 13
  - 1.3.2 Container stowage planning . . . 14
  - 1.3.3 Terminal operations . . . 16
- 1.4 Thesis contributions . . . 18
  - 1.4.1 The Cargo Mix Problem . . . 18
  - 1.4.2 The Flexible Ship Loading Problem . . . 20
- 1.5 Conclusions . . . 21
- 1.6 Future work . . . 22
  - 1.6.1 Improved modelling . . . 23
  - 1.6.2 Improved solution methods . . . 23

I Research papers 29

2 A Matheuristic for the Cargo Mix Problem with Block Stowage 31
- 2.1 Introduction . . . 32
- 2.2 Background . . . 33
- 2.3 Literature Review . . . 35
- 2.4 The Cargo Mix Problem . . . 37
- 2.5 Solution Method . . . 42
  - 2.5.1 Phase I . . . 43

2.5.2 Phase II . . . 45
2.5.3 Phase III . . . 46
2.6 Data Description . . . 47
2.7 Computational Results . . . 48
2.7.1 Intake optimisation . . . 49
2.7.2 Revenue Optimisation . . . 52
2.8 Conclusion . . . 53
2.A Stability Constraints . . . 56

3 Modelling and solving the Stochastic Cargo Mix Problem 61
3.1 Introduction . . . 62
3.2 Background . . . 63
3.3 Literature Review . . . 65
3.4 The Stochastic Cargo Mix Problem with Block Stowage (SCMPBS) . . . 67
3.5 Solution Method . . . 72
3.5.1 Phase I . . . 72
3.5.2 Phase II . . . 78
3.6 Data . . . 81
3.7 Computational Results . . . 82
3.8 Conclusion . . . 84
3.A Capacity Constraints . . . 86
3.B Stability Constraints . . . 87

4 Flexible ship loading problem with transfer vehicle assignment and scheduling 93
4.1 Introduction . . . 94
4.2 Relevant literature . . . 95
4.3 The Flexible Ship Loading Problem . . . 96
4.4 Mathematical Model for the Flexible Ship Loading Problem (FSLP) . . . 99
4.4.1 Enhancements for the FSLP model . . . 101
4.5 New lower bounds for the FSLP . . . 103
4.6 Heuristic approach . . . 105
4.6.1 Construction heuristic . . . 105
4.6.2 GRASP . . . 106
4.7 Computational analysis . . . 110
4.7.1 Data description . . . 111
4.7.2 Results for the mathematical model and enhancements . . . 111
4.7.3 Heuristic results . . . 115
4.7.4 Hierarchical vs integrated planning: value of integration . . . 118
4.8 Conclusion and future research direction . . . 120
4.A Container swapping: Speed up . . . 123

II Additional Work 125

5 Additional work for the Cargo Mix Problem with Block Stowage 127
5.1 Introduction . . . 127
5.2 Iterative method . . . 127
5.2.1 Improvement Step . . . 128
5.2.2 Evaluation Step . . . 129

5.2.3 Increasing the Neighborhood . . . 130
5.3 Phase I model . . . 130
5.4 Results . . . 132
5.4.1 Results for the Iterative method . . . 132
5.4.2 Results for the Phase I Model . . . 133
5.5 Conclusion . . . 134

6 Additional work for the Flexible Ship Loading Problem 135
6.1 Introduction . . . 135
6.2 Revised Flexible Ship Loading Problem model . . . 135
6.3 A Hybrid heuristic for the FSLP . . . 139
6.4 Column Generation for the FSLP . . . 142
6.4.1 The master problem . . . 142
6.4.2 The pricing problem . . . 144
6.4.3 The pricing problem as a shortest path problem . . . 145
6.5 Results . . . 148
6.5.1 Results for the R-FSLP model . . . 148
6.5.2 Results for the hybrid heuristic . . . 152
6.5.3 Column generation results . . . 153
6.6 Conclusion . . . 154

Chapter 1

Introduction

The emergence of container shipping has had a huge impact on today's society. Container shipping has changed the world through globalisation, and recent research shows that "*containers have boosted globalisation more than all trade agreements in the past 50 years put together*" (The Economist, 2013b). BBC cleverly describes a container, as "*the simple steel box that transformed global trade*" (BBC, 2017).

Liner shipping is the industry of shipping containerised goods using high-capacity ocean-going vessels, operating on regular routes with a fixed schedule. Within the industry, there is a fierce competition between the different actors, and as a result, container freight rates are historical low (UNCTAD, 2016). In search of economies of scale, liner companies are ordering bigger and bigger vessels. The capacity of these mega-vessels now exceeds 18,000 containers, and the growing fleet leads to planning problems in the industry. This thesis studies two of such problems with potential for minimisation of operating costs.

Figure 1.1: Global shipping traffic. Source: Shipmap.org (2016)

But, how did the container get to play such an important part in transforming global trade?

On April 26, 1956, the old oil tanker *Ideal X* embarked on its journey from Port Newark in New Jersey to Port of Houston, Texas. *Ideal X* had been refitted and strengthened to accommodate 58 containers, and is today known as the first commercially successful container ship. The owner of *Ideal X*, Malcolm McLean is accredited with revolutionising international trade and being the father of containerisation.

Before 1956, most cargoes were loaded and unloaded manually by dockworkers. McLean recalls the moment that gave birth to the idea of the container: "*I watched them take each crate off a truck and slip it into a sling, which would then lift the crate into the hold of the ship. Once there, every sling had to be unloaded, and the cargo stowed properly. The thought occurred to me, as I waited around that day, that it would be easier to lift my trailer up and, without any of its contents being touched, put it on the ship*" (The Economist, 2001). The current process was time-consuming, and often ships spent longer time docked than at sea. Moreover, the process was unreliable, and a lot of the goods were stolen, which influenced the cost of insuring the cargo. (The Economist, 2013b)

The immediate impact of *Ideal X* was clear; the cost of loading a tonne cargo dropped from $5.83 to $0.16, more cargo could be transported and unloading time was cut by up to three weeks. However, not everyone was happy; Freddy Fields, a top official of the International Longshoremen's Association, was asked what he thought of *Ideal X* to which he replied "*I would like to sink that son of a bitch*" (PCB, 2017). The longshoremen began to strike, but there was nothing they could do, the future was now, and the demand for labour was decreasing.

The next big breakthrough came in 1968 when the International Standards Organisation, standardised the dimensions and features of the container. Professor Brian Slack of Montreal's Concordia University notes: "*In terms of containerisation, which became a global phenomenon, it would have had great difficulty in doing that were it not for the fact that the boxes that everybody adopted were of fixed dimensions. The key to that is that you could design a ship to fit exclusively those dimensions of boxes*" (Australian Broadcast Company, 2014). Also, trucks and railway connection could be optimised for these dimensions allowing for inter-modal transport.

Not only longshoremen and related unions struggled with this revolution. It was catastrophic for many ports. The ports were built to facilitate operations in the old manner, i.e. typical features were storage sheds and lots of berthing space. Now, instead of storage sheds, you needed space on the dock to store containers. Existing ports tried to change, but it proved more efficient to develop new terminals on new sites instead of refitting existing ports to accommodate the container vessels. (Australian Broadcast Company, 2014)

As a consequence of containerization, over a five year period (from 1965 to 1970), port productivity saw an 18-fold increase, and insurance costs a six-fold decrease from £0.24 per tonne to £0.04, and the number of loading ports in Europe decreased from 11 to 3. (The Economist, 2013a)

During the Vietnam War, the US. government turned to container shipping to getting supplies to its troops, and container shipping started to prove its worth at an international level. During the 1970s and 1980s the container shipping industry grew exponentially, and from 1973 to 1983

the number of *Twenty-foot Equivalent Units* (TEUs) carried saw a four-fold increase, from 4 million TEUs to 12 million TEUs. (World Shipping Council, 2013)

Except for a bad year in 2009, following the financial crisis in 2008, the number of TEUs carried has kept increasing. Figure 1.2 shows how the demand has grown from 1996 to 2016.

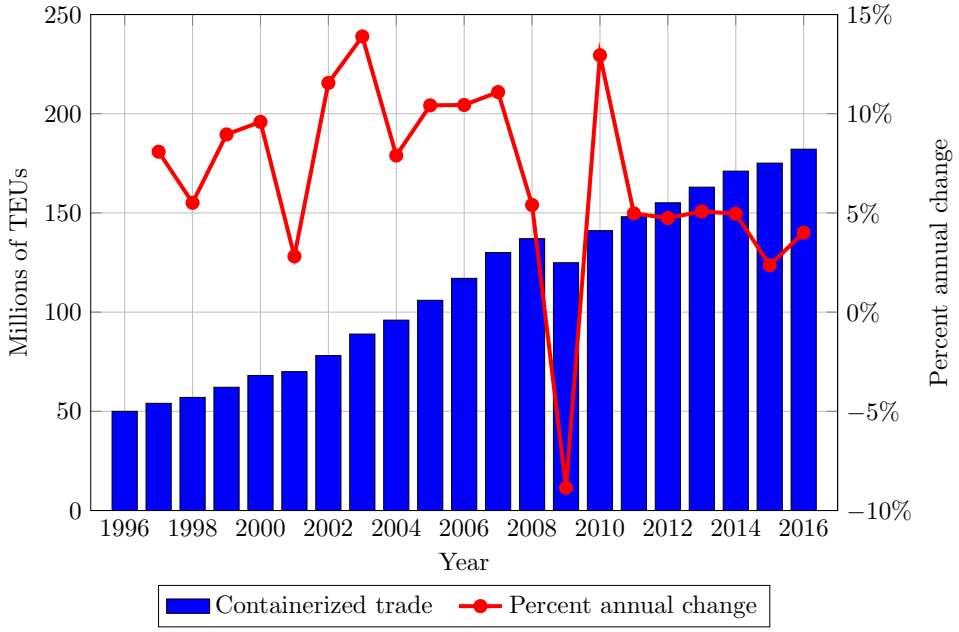


Figure 1.2: Global containerized trade, 1996–2016. Data Source: UNCTAD (2016)

With the increasing demand, container ships sizes have been growing accordingly. Over the last decade the average capacity of container ships has doubled, and as of May 2017 OOCL Hong Kong holds the world record for the largest containership, with a carrying capacity at 21,413 TEU. That is a factor 2.6 increase compared with the ∼8,200 TEU record set in 2003. It is expected that both the average and maximum size of containerships will grow over the coming years. With micro-optimisation in the design, new vessels could reach a 22,000 TEU capacity. Figure 1.3 shows how the maximum TEU capacity has developed since the 1970s. (OOCL, 2017; OECD/ITF, 2015)

Recently, slower-than-expected demand growth has resulted in historical low freight rates and surplus capacity. In 2015 the capacity was 7% larger than the demand, leading to 7% of overcapacity. In a recent analysis, The Boston Consulting Group expects this to significantly worsen, and perhaps double by the end of 2020 (Morley, 2016). The global container traffic is projected to grow between 2.2% and 3.8% annually from 2016 to 2020, this, however, is lower than the expected growth in capacity from ordered vessels, further increasing the oversupply of vessel capacity.

The report further notes *"in this challenging market context, companies that embrace digital will further strengthen their competitive advantage"* and continues *"smart use of technology can position the various players in the value chain (liners, terminals, intermodal players, customers) to communicate and collaborate more successfully."* Maersk Line must have reached the same conclusion, as they are positioning themselves to lead the digital transformation for liner shipping.

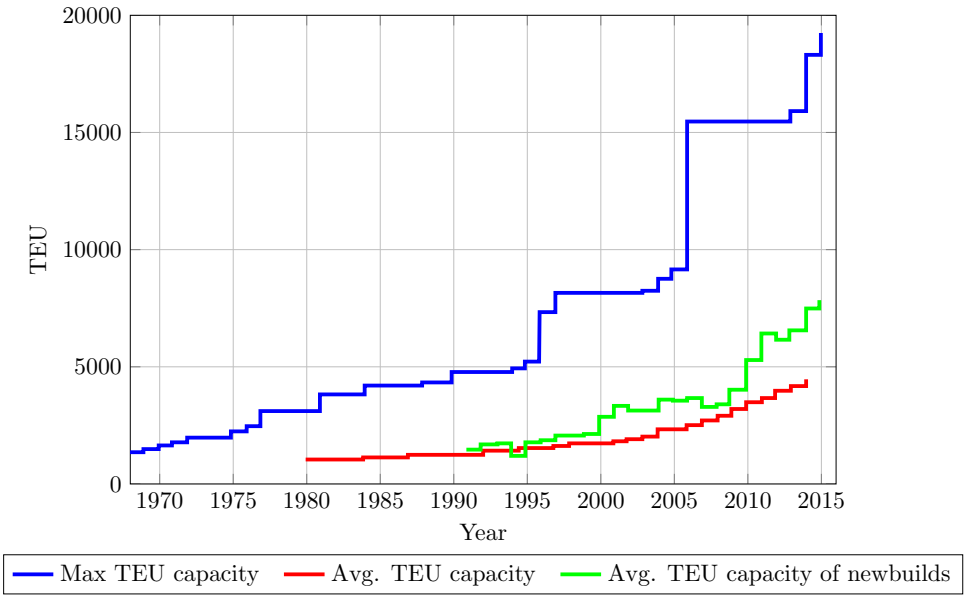


Figure 1.3: Development of container ship size. Data source: OECD/ITF (2015)

Over the coming years, they are installing sensors, bunker flow metres and updating communication technology one ship at a time. Niels Bruus, Head of Future solutions, fleet management and technology says: *"The short-term focus is to realise the efficiency benefits of more accurate, real-time data to optimize our operations. We expect that the impact of this data flow on our operational efficiency will be a significant positive"* (Maersk Line, 2016). The data will mainly be used to optimise operations and will thus mainly be used internally, but commercial opportunities will be considered in the near future (Maersk Line, 2016).

This thesis takes up this challenge and aims to provide insight into how data and optimisation methods can be used to improve operational efficiency within the domain of liner shipping.

1.1 Thesis outline

This thesis is divided into three main parts. First is this introduction to the thesis which gives a thorough presentation of the liner shipping industry, and the covered problems. The introduction also contains the scientific contributions, conclusions and ideas for future work. After the introduction, Part I contains the research papers written as part of the Ph.D. For two of the three research projects considered throughout the Ph.D. there are additional methods and results that have not been included in the research papers and are currently not submitted to any journal. These methods and the results thereof are described in Part II. The results are from new methods, and some of the results improve upon the state-of-the-art. With additional work, we believe some of these methods and their results can be the base of one or more standalone journal paper.

1.2 Introduction to maritime logistics

In the field of maritime logistics, you most often distinguish between three different modes of operation; industrial, tramp and liner shipping. This thesis focuses on planning problems within liner shipping. However, for the sake of completeness, we will in this section give a brief introduction to industrial and tramp shipping. After this, the domain of liner shipping is explained in greater detail.

1.2.1 Industrial and tramp shipping

Tramp shipping is similar to a taxi service. The ships follow the available cargo, and there is no published schedule. Often operators have a secure income from contracts, binding them to carry a specified quantity between specified ports within a given time frame. To supplement the contracts, tramp ships also trade on the spot market, where they can choose to accept optional cargo if they deem it to be profitable. The most common ships in tramp shipping are tankers and bulk carriers. A core planning problem faced by a tramp shipping company is to determine the split between spot cargo and fixed long-term cargo contracts and construct routes for the single vessels. When doing so, an estimation of future prices and demand must be taken into account as well as the fleet composition.

In industrial shipping, the operators most often control the vessels and the cargo shipped, and they strive to minimise the cost of shipping the cargo. Industrial shipping can be described as "own shipping", i.e. companies dealing with high enough volumes that it makes sense to undertake the shipping themselves. As with tramp shipping, industrial shipping mostly deals with liquid and dry bulk cargo that is shipped in full shiploads from their origin port to their destination port. Such cargo includes oil, coal, iron and chemicals.

Christiansen et al. (2007) presents an extensive introduction to maritime transportation and related optimisation problems faced in this field. This includes both tramp, industrial and liner shipping. The four review papers Ronen (1983), Ronen (1993), Christiansen et al. (2004), and Christiansen et al. (2013) give a thorough review of ship routing and scheduling problems within the field of operations research.

1.2.2 Liner shipping

Where tramp shipping resembles taxi services, liner shipping is often compared to buses. Similar to buses, liner vessels operate according to published itineraries and schedules. A round trip sailed at a fixed frequency is called a *service*, and the duration of a service usually determines how many vessels are assigned to it. For a service with a duration of 5 weeks, 5 vessels are generally assigned to ensure a specific departure day from each port. Services can usually be divided into a cargo intensive part, called the head haul, and a backhaul. On Europe - Asia services, the Asia to Europe leg of the service is the cargo intensive part as Asia, and especially China, produces many goods to be exported. The distribution network generally follow the trade, and is divided into intercontinental routes and regional routes. Intercontinental routes

transport goods to the desired region, hereafter the regional routes ensure the cargo is delivered at its desired destination port. Also, feeder lines are used to connect smaller ports with the intercontinental and regional routes.

The main cost components in liner shipping are vessel costs, cargo handling costs and bunker costs (the fuel consumed by container vessels). Stopford (2009) break down the major cost components and estimates that for an 11,000 TEU vessel operating costs account for 7% of the cost, bunker costs for 39%, port and canal fees for 10% and lastly capital costs for 43%. The operational costs include crew, maintenance and insurance and capital costs account for the financing of the vessels. The port and canal fees are the fees for calling a port and traversing canals. Cargo handling costs are not accounted for in the estimate. As the oil prices have been increasing, operators have started to sail their vessels at a slower speed, known as *slow steaming* in the industry. There is a cubic relationship between the speed of the vessel and the bunker consumption, so sailing slow will decrease the bunker consumption for a fixed distance. Newly ordered vessels are built with a lower design speed, benefiting more from slow steaming.

Revenue is obtained by transporting cargo, and as mentioned earlier the freight rates are highly volatile. With Europe importing more than they are exporting, there is an overflow of empty containers ending up in Europe and an imbalance in the trade. For this reason, it is usually cheaper to transport a container from Europe to Asia, than the other way around. The liner accepts cheaper rates from Europe to Asia, as they will not need to ship as many empty containers to where they are needed.

Liner shipping companies organise themselves in alliances. By doing so, all actors in an alliance pool together their fleet of vessels, and enters vessel sharing agreements. They do so as no carrier have enough vessels to offer weekly sailings across every port they serve. Two or more carriers share a service, and they each deploy a number of vessels ensuring a high frequency. Each carrier will operate their share of the vessels and have a corresponding share of the capacity on the other vessels on the same service operated by their partners. Thus each member of an alliance can offer a better product for customers at the same operating cost. For smaller carriers, this means that they have a greater geographic coverage than they would otherwise.

For liners, it is a challenge to get precise demand forecast. For the customer, there is no fee for booking cargo, and they will only pay for a container transport once it is undertaken by the liner. Often customers will book more cargo than what they actually will need at the time of departure; this is to ensure enough slots on the vessels. The carriers know this, and thus expects a certain amount of *no-shows*.

The main infrastructure needed to facilitate efficient and cost-effective transportation of containers include the liner vessels, container terminals, canals and of course the containers themselves. We will describe this in the following.

1.2.2.1 Containers

The intermodal freight container comes in different sizes and with varying capabilities. Figure 1.4 shows the dimensions of the most common containers. *20 foot containers* are 20' long, 8' wide and 8.6' high. The standard size of a *40 foot container* only varies in the length compared to a 20'

container. However, *40 foot high cube containers* are 1' higher than the standard size. Containers with length 45', 48' and 53' also exists, but they are less common. Container capacity is expressed in *Twenty-foot Equivalent Units* (TEUs), and one standard 40' container corresponds to 2 TEUs.

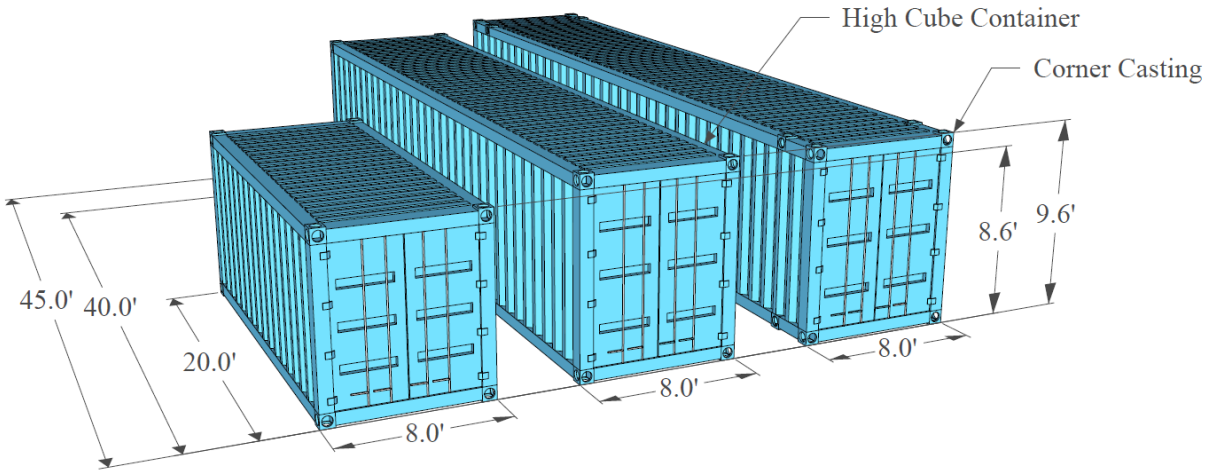


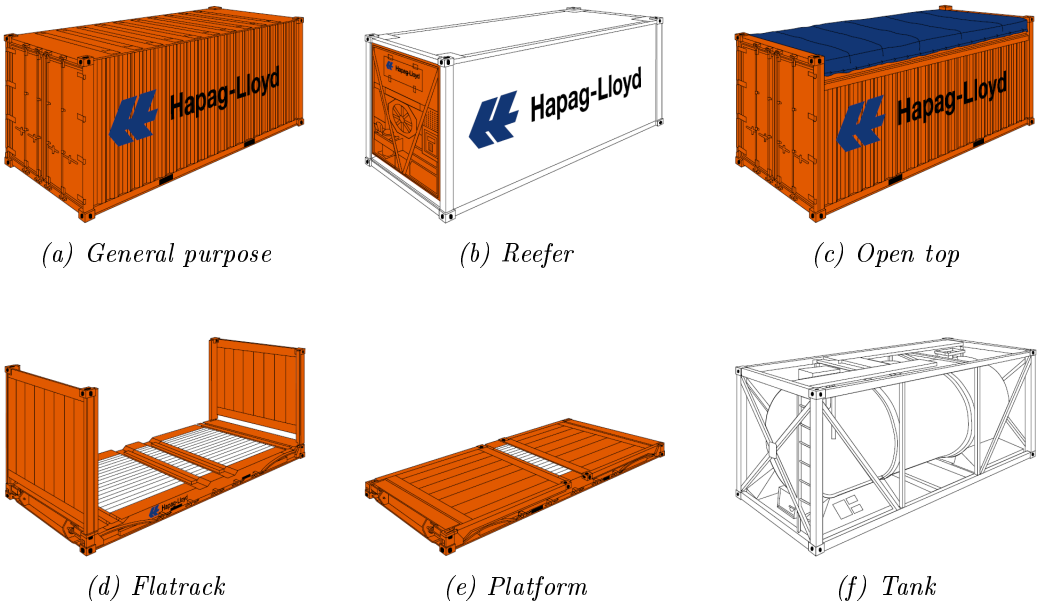
Figure 1.4: Dimensions of the most common ISO standard containers. Source: Pacino (2012)

The corner castings in the corners of the containers are used to lock containers securely together. The containers are structurally enforced in the corners, and for this reason, 20' containers cannot be stacked above 40' units, at the risk of the container below collapsing under the weight. As seen in Figure 1.4, 45' containers also have corner castings at the 40' position, ensuring they can be stacked on top of a 40' container.

Figure 1.5 shows the different types of container. The general purpose container (Figure 1.5a) is the most common. Refrigerated containers (or reefers) as seen in Figure 1.5b are temperature-controlled containers and can be used to transport perishable goods, e.g. fruits, vegetables, but are not limited to foodstuff. Reefers are equipped with a cooling unit but need to be plugged into an electrical outlet on the ship to get power. The top of an open top container (Figure 1.5c) is covered by a tarpaulin instead of a fixed roof. This allows oversized goods (in relation to the door opening) such as timber and metal waste to be loaded from above. Flatrack containers (Figure 1.5d) are particularly suitable for heavy loads and heavy goods. Platform containers (Figure 1.5e) can be used for heavy loads and oversized length cargo. Tanks (Figure 1.5f) are used to transport liquid and gasses, for example, foodstuff (alcohols, fruit juices, edible oils) or chemical products, e.g. flammable material and toxic substances. When stowing dangerous goods containers, certain separation requirements must be obeyed.

As mentioned, there is an imbalance in the trade, and empty containers are being shipped to where they are needed. This is costly, as an empty container generate no revenue, but still take up capacity on the vessel. For this reason, collapsible containers are being developed. These containers can be collapsed when being empty, and four collapsed containers take up space as a regular container.

Containers are built to support standard shipping, storing and handling conditions, and they can be used multiple times. The lifespan of a container depends on the usage level, and the conditions they are exposed to. The lifespan usually ranges from 10 to 15 years.



(a) General purpose *(b) Reefer* *(c) Open top*

(d) Flatrack *(e) Platform* *(f) Tank*

Figure 1.5: Type of containers. Copyright: Hapag-Lloyd AG. Source: Hapag-Lloyd (2017).

1.2.2.2 Liner vessels

The worldwide fleet of liner vessels (or container vessels) consists of approximately 6,000 ships, with a total capacity over 21 million TEU, and the largest having a capacity over 21,000 TEU. (Alphaliner, 2017; OOCL, 2017). Despite the size of these vessels, they can be operated by a crew of around 20 members. Containers vessels are classified according to their size and capacity into the following main groups.

Small Feeders ($\leq$ 1000 TEUs): Small feeders are usually used for short-sea shipping and might be outfitted with cargo cranes. This allows them to serve small ports where it is not economically viable to invest in cranes.

Feeders (1000 – 2800 TEUs): These vessels are most often used to feed very large vessels, or complement the small feeders in servicing markets that are too small for larger vessels.

Panamax (2800 – 5100 TEUs): These are the vessels that fulfil the requirements for travelling through the original Panama Canal opened in 1914.

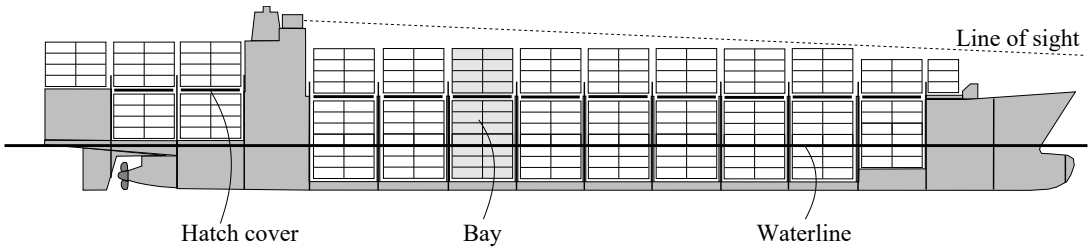
Post-Panamax (5100 – 10000 TEUs): These are the vessels that exceed the original Panama canal beam.

New Panamax (12000 – 14500 TEUs): In 2009 the Panama Canal Authority published specifications which would come into effect when the new Panama Canal opened in 2016. These are the vessels that can sail through the new Panama Canal.

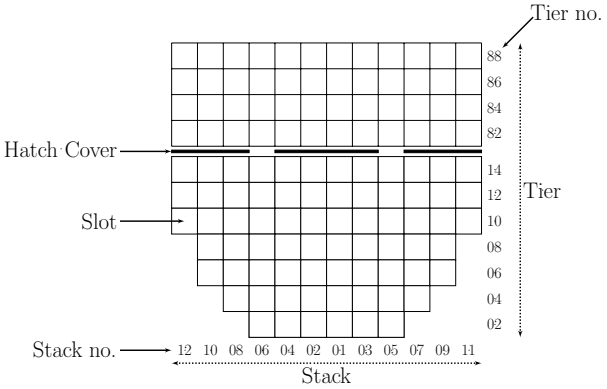
ULCV ($\geq$ 14500 TEUs): Ultra Large Container Vessels (ULCVs) are the vessels already larger than the new Panama Canal, but they are still able to transit the Suez Canal.

The layout of a container vessel is shown in Figure 1.6. As seen in Figure 1.6a the vessel is longitudinally divided into sections called *bays*. A single bay is further organised into two

sections, on-deck and below-deck, separated by a metallic leak-proof structure known as a *hatch cover*. Figure 1.6b shows the layout of a single bay, and the standard indexing system used in the industry. Bays are divided in *stacks* (horizontal index) and *tiers* (vertical index). A vertical position in a stack is called a *cell* consisting of two *slots*. A cell usually has a capacity of 2 TEUs, meaning either two 20' containers or one 40' container can be stowed. Containers below-deck can only be accessed once all containers on top of the hatch covers are removed as well as the hatch itself.



(a) Example figure of a container vessel. Source: Christensen and Pacino (2017)



(b) Bay layout. Inspired by: Kim et al. (2004)

Figure 1.6: A container vessel

Liner vessels are equipped with carefully located ballast tanks along the length of the vessels. They can be used to change displacement and stability conditions of the vessel, e.g. by allowing for ballast water to be pumped out to temporarily reduce the *draft* (distance between the surface of the water and the lowest point of the vessel) of the vessel when entering shallow water. Some tanks are placed on the sides of the vessels and can be used to stabilise the vessel during (un)load operations in port.

1.2.2.3 Container terminals

A container terminal consists of a quayside, a yard, and a landside area. Looking from the sea, the first thing you will notice is the quayside with *quay cranes* towering above the ships. These immense structures are used to load and unload the vessels and constitute the backbone of any container terminal. Most quay cranes are built on tracks in the surface, making it possible to

move left and right parallel to the berthed vessel. The berth is the place where the vessels moor, and is the interface between the vessels and the terminal. (Steenken et al., 2004)

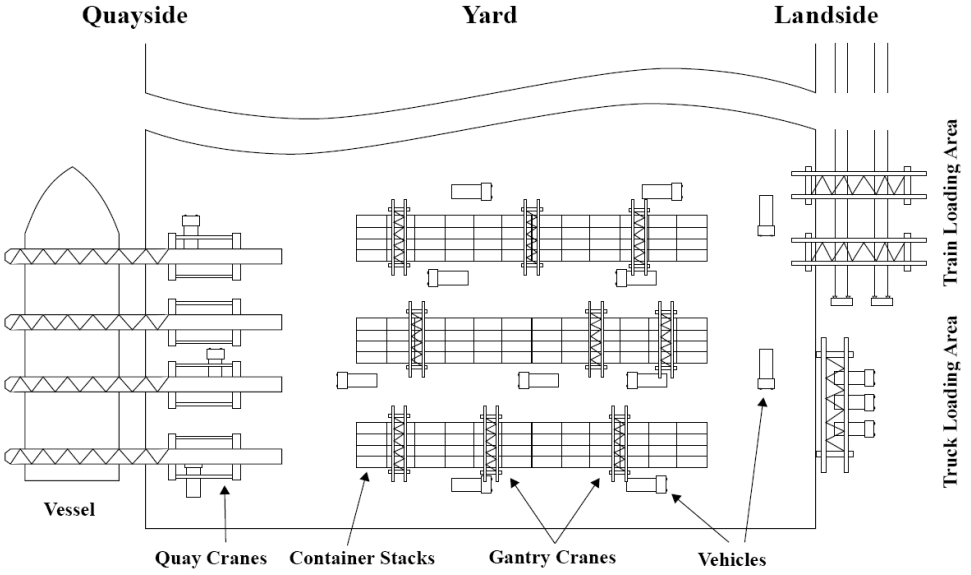


Figure 1.7: Container terminal layout. Source: Pacino (2012)

The quay cranes are the main bottleneck of most container terminals and to serve them, trucks are endlessly arriving and stopping beneath the cranes in precise locations. Either the cranes are picking a container of the trucks, or they are dropping a newly unloaded container to be stored in the yard. Different types of transport vehicles exist, from the unmanned *Automated guided vehicles*, to the more standard *yard truck* and *straddle carrier*. Straddle carriers can lift and stack containers, and can thus be used to avoid crane waiting time, as containers can be unloaded on the ground, leading to increased crane productivity. The different transport vehicles available are described in greater detail in Meisel (2009) and Steenken et al. (2004)

The yard area is used for temporary storage of containers. Here the containers are organised in multiple blocks, each consisting of several parallel container rows and a number of lengthwise positions. Multiple tiers of containers can be stacked at each position. In the yard, containers are usually stacked five containers high, and *rail mounted gantry cranes* (RMGC) serves as input/output devices to store and retrieve containers from blocks. An RMGC spans up to 13 container rows. Depending on the design of the terminal, either one of the rows is reserved for serving transport vehicles, or vehicles are served at the front side of a block. The first alternative requires less movement of the gantry crane, whereas the second allows for higher storage capacity. Not every terminal use RMGCs, terminals using straddle carriers in the yard area also exists (see Meisel (2009)).

In the yard, there are often specific storage areas for export, import, special and empty containers. A container maintenance area is also located in the yard. The maintenance area is used for washing or repairing damaged empty containers. Some container terminals also have *sheds* where containers are (un)packed and goods can be stored.

At the landside area, containers are (un)loaded on/off trucks and trains, and thus provide a

connection to the hinterland. In order to ensure efficient and effective low-cost transportation of containers for inland customers, many terminals are connected with railway tracks leading into the terminal, allowing the terminal to serve trains with the gantry cranes. The capacity of one train is about 120 TEU. Trucks can enter the terminal through the gatehouse. They need to pass a security check and have their transport document checked. After this, they are sent directly to a designated area where they will be served. Trucks can have a capacity of up to three TEU. No trucks entering the gatehouse are allowed to access the quayside. When a container arrives at the terminal (whether from quayside or landside operations), it is taken to the yard, and from here another transport vehicle will take it to the next step on its journey when needed.

1.2.2.4 Canals

Transporting goods by sea from the far east to the west can take several weeks, but that would be even larger had it not been for the Panama- and the Suez Canal.

The Suez Canal connects the Mediterranean Sea to the Red Sea and is one of the worlds busiest shipping lanes. The 193km stretch of water allows ships to travel between Europe and the Far East, without sailing around Africa reducing the sea voyage distance by approximately 7000km. The maximum beam (width) of a vessel passing through the Suez canal is approximately 80m, but the canal is only opened in one direction at a time, enforcing vessels to wait upwards of 22 hours. With the *New Suez Canal*, opened in 2015, the waiting time was cut down by more than 50%, doubling the daily capacity. The Panama Canal connects the Pacific Ocean with the Caribbean Sea and the Atlantic Ocean. The strong winds, strong currents, large waves and icebergs make the Cape Horn route some of the most treacherous waters in the world to navigate. By using the Panama Canal, ships can avoid navigating around Cape Horn, while also reducing the length of the route by around 4800km. However, only vessels with a TEU capacity of max. 14500 TEUs can pass through the Panama Canal. Another canal that must be mentioned in this context is the Kiel Canal, making for easy access from the North Sea to the Baltic Sea, avoiding sailing around the Jutland peninsula. (Velta, 2016)

While it is costly to transit these main canals, the costs are offset by the benefits. The shorter transit times allows ships to be utilised for more sailings per year, while also delivering a better product for the customers. These canal systems also have a tremendous impact on the economy in the surrounding community. In Egypt, the Suez Canal contribute 2 percent to the economy, while it is 6 for the Panama Canal. (Dupzyk, 2015)

While the Suez Canal is used for almost all Europe-Far East connections, the picture is a little more blurry for the Far East-US connections. The length of Far East-Suez Canal-US East-coast journey is only 5% longer than the Far East-Panama Canal-US East-coast counterpart. As the Suez Canal can accommodate larger vessels, some liners prefer the Suez Canal. Also, when fuel prices are low, the Suez is more beneficial compared to when the fuel is expensive.

Recognising the economic benefits, the Nicaraguan parliament has approved plans to build a Nicaragua Canal. However, with the uncertain status, it might not be built, but competition could get tougher over the next 15 years. Also, there have been discussions for a Thai Canal set to divert traffic from the Strait of Malacca between Malaysia and Indonesia. However, there are no concrete plans.

1.3 Liner shipping and operations research

The global container transportation system has reached a size where humans no longer easily can comprehend the complexities and assess the best configurations. Therefore the need for advanced decision support tools is increasing.

Beginning with the survey Ronen (1983), a survey of research on ship routing and scheduling (and related) problems have been published every decade (Ronen, 1983; Ronen, 1993; Christiansen et al., 2004; Christiansen et al., 2013). These surveys provide a comprehensive study of published scientific work on ship routing and scheduling. The surveys mainly focus on the work published since the last survey, and can thus also be used to reveal trends in research activities within the domain. In general, the number of newly published research papers about doubles every decade. Specific for liner shipping, the number of published papers in the 5-year period from 2007 to 2011, saw a fast increase compared to the five years before. The lack of research on liner shipping is, therefore, closing fast. (Christiansen et al., 2013)

The survey "Optimization in liner shipping" (Brouer et al., 2017) gives an overview of optimisation problems faced by the liner company. The problems are classified according to the strategic, tactical and operational level as seen in Figure 1.8. The most determinative strategic problem is to determine which markets to serve. Hereafter, the fleet size and mix can be established, which all provides input to the network design problem. As previously mentioned, bunker consumption grows cubically with speed. Determining the sailing speed on a leg (Speed Optimization), is, therefore, an important tactical problem, where the bunker cost must be considered while ensuring timely delivery for the customers. The cargo routing problem determines how the cargo should be routed through the network once it has been accepted. Due to the imbalance in the trade, containers are in short supply at some locations. The empty repositioning problem is a more operational problem, and concern using residual capacity in the network to provide empty containers where they are needed. Due to severe weather conditions and delayed terminal operations liner vessels are often delayed. Disruption management problems provide insight how to handle these disruptions efficiently. For a comprehensive review and classification of the OR literature related to liner shipping, we refer to Meng et al. (2014).

The mentioned problems are often highly interdependent. The empty repositioning depends on the cargo routing which depends on the network design. Instead, of treating this problem as a single problem, they are often treated independently. Additional costs savings could be gained by solving the combined problem. However, solving the problems hierarchically already poses enough challenges, and can still provide valuable business insight.

In this section, different planning problems faced by the liner shipping industry are described in further details. We do not limit ourselves to problems faced by the liner company, but also provide an overview of terminal-side planning problems. The purpose is not to give a complete overview but mainly serve as an introduction to the details and considerations that must be accounted for.

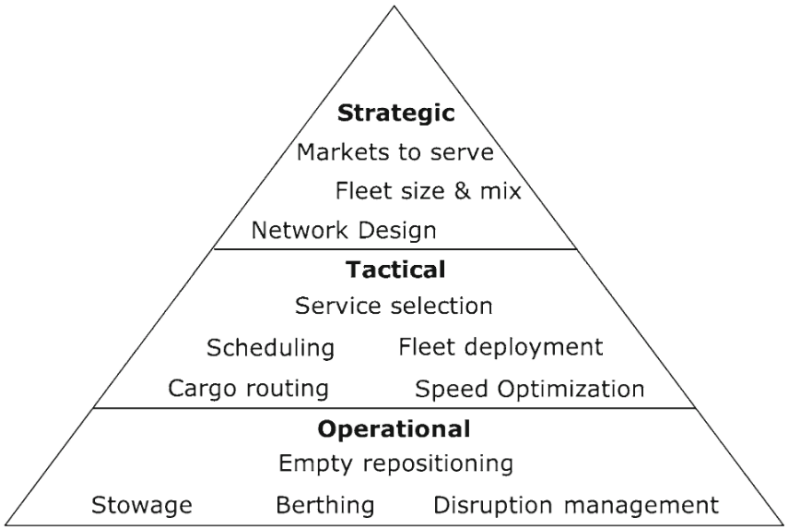


Figure 1.8: Main planning problems in liner shipping. Source: Brouer et al. (2017)

1.3.1 Network design

Liner ships operate along published itineraries and schedules. The frequency, timetables and fleet deployment, may change depending on demand fluctuations, but the routes themselves rarely change. Therefore the route network design is an important strategic decision. The routes in the network are identified by the ports they visit, and the order. Additionally, the size of the ships assigned to a route must also be determined. The problem of constructing routes to serve is referred to as The Liner Shipping Network Design Problem (LSNDP).

Most modern liner shipping networks are composed in a way similar to a hub-and-spoke network, with *trunk services*, connecting hubs from different geographical regions to each other. *Feeder services* are the spokes of the network, and are mostly contained within a specific region to serve a main port and a set of smaller ports that do not merit visits by the trunk services. Due to the structure of the network, only a few direct connections exist (relative to the size of the network). Instead, a container can be unloaded, and temporarily stored, at an intermediate port until it is loaded on another ship to reach its next destination. This is called a transhipment and containers often have 1 or 2 transhipments on its journey to the destination port. For every transhipment, there is an associated handling cost, and the number of transhipments should thus be kept to a minimum.

Related to the LSNDP is the Fleet Deployment Problem. Compared to the LSNDP the fleet deployment problem is a more tactical problem, where the network is considered fixed and vessels are assigned to routes in the network. A liner shipping company often has some flexibility by in- or out chartering vessels. If the demand on a specific trade route is over-estimated, a liner shipping company might be able to profit by out-chartering vessels. The fleet deployment problem aims to serve all routes in the network at a minimum cost while ensuring the demand is fulfilled.

1.3.2 Container stowage planning

A container stowage plan is the allocation of containers to specific slots on a vessel. Stowage plans are made by *stowage coordinators* at the liner shipping company. They are used in the terminal to coordinate the load and discharge operations, and thus the stowage coordinator must complete the stowage plan before the vessel calls into port.

When making a stowage plan, the data available for stowage coordinators include; a *loadlist*, information about the vessel condition, port information, and a forecast mainly based on historical data. The *loadlist* contains information concerning the containers to be loaded. This includes information such as weight, height, type, and the discharge port. The vessel data used includes the status of ballast and fuel tanks, as well as information (height, weight, type and position) of the containers on the vessel.

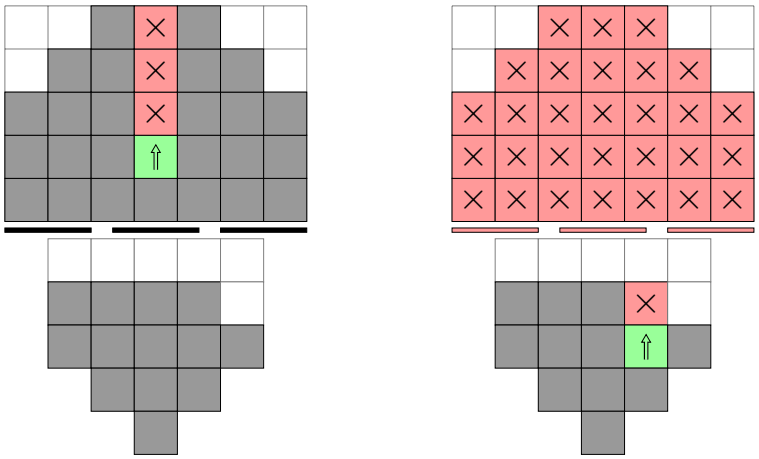
Stacking rules impose how the containers must be stacked. As previously mentioned, no 20' containers can be stacked above 40' units. Also, reefers must be stowed near an electrical outlet to get power to the cooling unit and dangerous goods containers must be stowed according to a complex set of separation rules. Normally the weight of containers must decrease upwards in stacks on deck.

A stowage plan must result in a seaworthy vessel. The captain has the responsibility to declare the vessel seaworthy. Besides ensuring that the vessel is not defective or undermanned, this also entails that all container stacking rules are satisfied, and special containers are handled correctly. The static stability needs to be correct, and all stress forces must be within limits. The weight of the cargo causes shearing and bending stresses over the vessel structure, which must be within limit at certain calculation points. Ensuring the vessel is stable also requires the centre of gravity to be within defined limits. The *aft draft* is the distance between the waterline and the bottom of the hull at the stern of the ship, and the *fore draft* is at the bow. The trim is defined as the difference between the aft draft and the fore draft. The trim, fore draft, and aft draft must all be within limits. To safely navigate, the captain standing on the bridge must be able to see 500 meters ahead of the ship, or the double of the length of the vessel (whichever is smallest). This *line of sight* requirement limits the height of a stack on-deck. Additional stability requirements are described in Delgado (2013) and Pacino (2012).

All data concerning the vessel structure is described in a document called *the vessel profile.* The vessel profile describes the location of the tanks as well as their capacities. It also contains the weight and capacity limits as well as the *hydrostatic table.* The hydrostatic table is used to calculate draft, the centre of gravity and trim.

There might be many stowage plans that result in a seaworthy vessel, and it is the stowage coordinators responsibility to find one that is economically viable as well, while also making sure the vessel can be fully utilised at downstream ports. As previously mentioned, port costs are a major cost component of liner shipping, so a good stowage plan is one that helps to minimise the time at port and secondary reduce fuel consumption. Two main features impact the time at a port; crane split, and overstowage. Together with the port information, the stowage coordinator receives information regarding the number of quay cranes assigned to the vessel, and they will try to distribute the containers evenly along the length of the vessel, to allow for multiple cranes to have a similar workload. *Overstowage* leads to extra crane moves, and should thus

be minimised. There are two kind of overstowage, *stack overstowage*, and *hatch overstowage*. Figure 1.9a illustrates a bay where we have to unload the container with the arrow (green). To retrieve this container the containers with the crosses (red) needs to be moved as well. If these containers do not need to be unloaded at the current port, this leads to extra cranes moves. Figure 1.9b illustrates hatch-overstowage. Here all containers on-deck needs to be moved, before removing the hatch covers and the single overstowing container below-deck. Concerning fuel savings, the water in the ballast tanks should be minimised. Ballast water can constitute up to one-third of the vessels total weight, thus minimising the ballast water can have a drastic impact on the fuel consumption. Additionally, stowage coordinators aim to distribute the weight of the containers along the length of the vessel such that the trim is close to optimal and thereby minimising the drag.



(a) Stack overstowage *(b) Hatch overstowage*

Figure 1.9: Examples of different kinds of overstowage.

The academic work on stowage planning problems is challenged due to a missing common problem definition. The number of container types considered varies from paper to paper, as does the importance with respect to stability considerations and the seaworthiness of the vessel. This also means there is no unified benchmark set for which all solution approaches are tested, thus making it hard to compare methods to each other. Businesses in the industry are very reluctant to share real-life vessel data. Thus confidentiality issues challenge the existence of a unified benchmark. One could try to generate a vessel profile themselves, but without a real hydrostatic table, it is unsure if the benchmark will be representative of a real-life vessel.

Due to its similarity to work presented in this thesis, we find it relevant to review the academic literature on stowage planning problems. The literature can be divided into two main groups, Single model approaches and multi-phase methods. The first full detailed mathematical model is presented in Botter and Brinati (1992). Additional approaches based on Mixed Integer Programming formulations are the work of Ambrosino et al. (2004) and Li et al. (2008). These models experience scalability issues due to the large number of variables, thus making them unsuitable for practical use. Heuristic single model approaches include the placement heuristic by Aslidis (1984) and the *Suspensory Heuristic* by Avriel et al. (1998). The suspensory heuristic only focuses on container stacking constraints and does not include any stability considerations. Dubrovsky et al. (2002) obtain results similar to the suspensory heuristic but has the possibility of incorporating simple stability constraints. Ding and Chou (2015) describes a state-of-the-art

heuristic that improves the results from the suspensory heuristic. The placement heuristic in Low et al. (2009) is based on the work practice of stowage coordinators, and the similarities to packing problems are explored in Sciomachen and Tanfani (2003), where a 3D-bin packing heuristic is used to tackle a stowage problem. Constraint programming approaches worth mentioning for stowage planning includes the work of Ambrosino and Sciomachen (1998).

The first multi-phase method was proposed in Botter and Brinati (1992). However, it was too computationally expensive. Instead, the problem was solved by a branch-and-bound search. The first multi-phase method that presented promising results was Wilson and Roach (2000). Here, first containers are assigned to general areas on the ship following high-level capacity constraints, known as the master bay planning problem. Hereafter the specific containers are assigned a specific slot (slot planning problem). Kang and Kim (2002) uses the same decomposition but in an iterative manner. Pacino et al. (2011) solves industrial scale problems using a mixed integer programming approach for the master bay planning problem, and a mix of a constraint programming and a constraint-based local search method for the slot planning problem. A three-step heuristic is introduced in Ambrosino et al. (2009), in which the left and right, and bow and stern weight differences are kept within a tolerance. The slot planning problem has also received attention. The GRASP approach presented in Parreno et al. (2016) is the current state-of-the-art. The constraint programming methods of Delgado et al. (2012) and Pacino and Jensen (2012) must also be mentioned in this context.

Pacino (2012) provides a more in-depth overview of the literature for both single model approaches and multi-phase methods.

1.3.3 Terminal operations

A container terminal faces planning tasks on all levels of the organisation. Figure 1.10 is an overview of terminal planning problems grouped according to the organisation planning level (strategic, tactical and operational) and where the planning problem is located at the terminal (seaside, yard or landside). The figure also shows the underlying problem interdependencies. In the following, a few selected operational problems are described in greater detail. For a more thorough summary of these problems see Meisel (2009).

Before vessels are moored at the terminal, it must be assigned a berth. In the Berth Allocation Problem (BAP) a set of vessels to be served within a planning horizon are considered. These vessels might have different lengths including clearance and draft. A feasible berth-plan is one where all vessels are given a berthing position and berthing time, ensuring they do not occupy the same quay space at a time, and they fit within the boundaries of the quay. A good berth-plan is one that provides fast and reliable service of vessels. In the literature, this is often represented by minimising the port stay times, i.e. the sum of waiting and handling times.

Beside the quay space, the quay cranes (QCs) are a scarce resource at a container terminal. Therefore, after the vessels have been assigned berths, the Quay Crane Assignment Problem (QCAP) aim to assign QCs to vessels optimally. For each vessel, the number of containers to be loaded and unloaded is known, and the QCAP assigns QCs to vessels ensuring that all containers from/to the vessel can be (un)loaded. The length of the vessel and the safety distance between the QCs defines the maximum number of QCs that can work on the vessel at a time, while

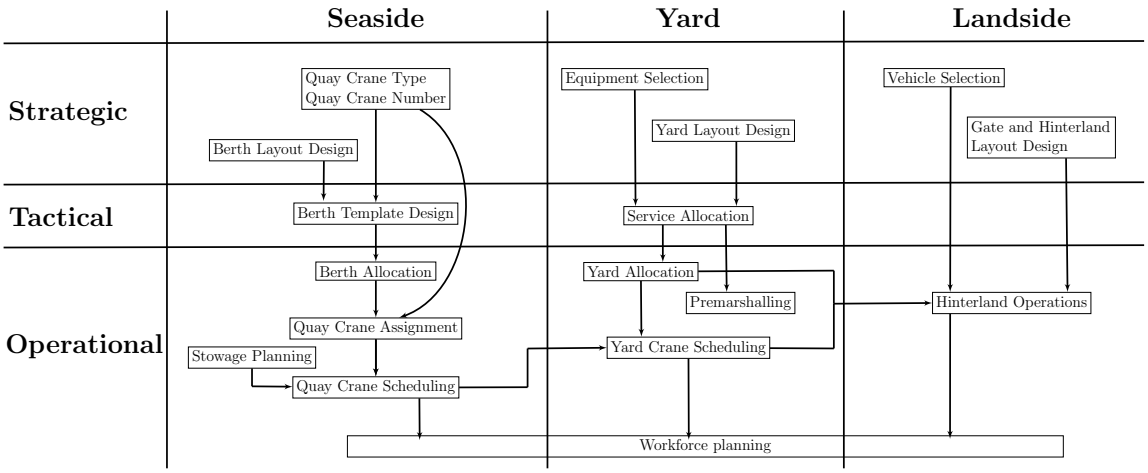


Figure 1.10: Planning problems in container terminals. Inspired by: Meisel (2009) and Iris et al. (2016)

contracts (if any) determines the minimum. The QCs can move alongside the quay, but cannot overtake each other. Therefore, the QCAP both determines the number of QCs assigned, and which specific QCs that make up the full assignment.

In connection with the QCAP is the Quay Crane Scheduling Problem (QCSP). Given a quay crane assignment and the *tasks* associated with each QC, a solution for the QCSP defines the order in which the tasks will be processed and the starting times. The tasks are the workload for the vessel and can either be general bay areas to be (un)loaded, a bay, a stack or down to (un)loading a specific container. Precedence relation between tasks ensures that unloading operations are handled before loading and that the schedule complies with the stowage plan. For a QC schedule to be feasible, all tasks must be processed, and a QC must only process one task at a time. In the QCSP the minimum makespan is usually preferred, as this corresponds to the turnaround time for the vessel.

These three problems are traditionally solved in a hierarchical fashion, where the solution to the BAP is used as input to the QCAP, which is then given as input to the QCSP. The underlying assumption here is that a good solution to an upper-level problem will lead to a good solution to a lower-level problem. However, this is not always the case, and this leads to suboptimal decision making. Instead, problems are starting to being considered in a more integrated fashion, where details from more than one problem are being considered together. These problems are usually harder to solve, and it can be hard to capture all the details of the specific problems. Integrated problems seen in the literature includes Berth Allocation and quay Crane Assignment Problem (BACAP), Berth Allocation and quay Crane Assignment and Scheduling Problem (BACASP) and Quay Crane Assignment and Scheduling Problem (QCASP). The QCSP is also often integrated with yard management problems.

Yard management problems include designation of partial yard blocks as buffering areas for the import and export containers for calling vessels. To minimise travelling times, the designated space should be close to where the vessel is expected to berth, but should also be spread among the gantry cranes. Additionally, there is the premarshalling problem, where idle times for automated gantry cranes are used to order the containers in a block. The containers are ordered according

to at what time they should be retrieved, thereby utilising idle time to minimise waiting time that might occur at a later point in time.

At the tactical level are problems in the yard related to storage of empty and reefer containers. These decisions will last for weeks, if not months. The strategic problems mostly concern the location and layout of the terminal as well as the type of equipment to invest in.

1.4 Thesis contributions

This thesis considers two planning problems that to some extent currently are understudied in the operations research literature. The thesis thus contributes by formally describing the problems and introducing them to the OR literature. For both of these problems, novel solution methods are described and compared with standard techniques. Thus this thesis both contributes methodologically and computationally. In the following, we introduce the studied problems and outline the scientific contributions for both of them. An overview of the dissemination of the work is also given.

1.4.1 The Cargo Mix Problem

Container vessels constitute an important part of the backbone of international trade. In search of economies of scale container shipping lines are building bigger and bigger vessels. Comparing modern ∼19,000 TEU vessels with older ∼15,000 TEU vessels, there are large savings to be gained. A recent report (OECD/ITF, 2015) estimates that between 55% and 63% of this savings is due to technological advances and especially design changes. With the growing bunker prices, modern vessels are being optimised to sail at a lower speed to save costs. Hence, the relative savings will vanish over time as new medium-sized vessels are being built. (Berglund, 2017; OECD/ITF, 2015)

For large vessels, it is crucial to utilise their full capacity to sustain being cost effective. E.g. an 18,000 TEU vessel requires a 91% utilisation to be cost-effective over a fully loaded 14,000 TEU vessel (OECD/ITF, 2015). Assuming there is enough demand, it is hard to achieve the maximum capacity due to, e.g. container weight distribution. The weight distribution of the vessels plays an essential role for the seaworthiness and the safety on board.

Most strategic and tactical decisions in liner shipping are based on the nominal capacities of the vessels. However, unless the cargo weight distribution is perfect, the nominal intake cannot be reached. The focus of the cargo-mix problem is to analyse the cargo composition needed for a vessel to maximise its utilisation on a given service. By cargo-mix, we refer to the composition of cargo in terms of container types and weights. Such a model can have multiple applications ranging from driving freight rates, and improving the vessel capacity model in fleet deployment and network design problems. Aside from strategic applications, it can also be used as an analytical tool to perform various what-if analyses.

Chapter 2, Chapter 3 and Chapter 5 all deal with the Cargo Mix Problem. We use real-life

vessel data provided by our industry collaborator. In Chapter 2 we consider the cargo-flows (origin-destination container-demand matrix) as being deterministic and known in advance. We impose a high degree of accuracy concerning stability considerations to ensure the method is industrially applicable. The existing definition of the problem is extended to include a specific stowage strategy used in the industry (block stowage), circular routes, draft restrictions and cargo flow limitations. We show that with the extensions, the problem cannot be solved using standard MIP solvers. The main reason for this is the inclusion of the stowage strategy. Instead, a matheuristic multi-phase approach is used. In the first phase variables related to the stowage strategy are heuristically set. Hereafter, the next phases assign containers to general areas on the ship ensuring the decision in the first phase is fulfilled. The results show that the matheuristic is scalable and able to find quality solutions for real-life instances within a matter of seconds. This paper contributes to the state-of-the-art by; 1) Extending the existing definition of the problem, 2) extending the existing compact formulation of the problem including the new features, 3) propose and compare a number of matheuristics that can solve the problem in a matter of seconds. The work has been disseminated as follows:

- A paper published as *J. Christensen and D. Pacino (2017). "A matheuristic for the Cargo Mix Problem with Block Stowage". In:* Transportation Research Part E: Logistics and Transportation Review *97, pp. 151–171.* ISSN*: 1366-5545* (Christensen and Pacino, 2017).

- Presentation by Dario Pacino at *Sixth International Workshop on Freight Transportation and Logistics (Odysseus 2015)* (Christensen et al., 2015b)

- Presentation by Jonas Christensen at *the hEART 2015 conference* (Christensen et al., 2015a)

- Presentation by Jonas Christensen at *9th Triennial Symposium on Transportation Analysis (TRISTAN 2016)* (Christensen and Pacino, 2016)

In Chapter 5 two extensions to the existing heuristic is presented. Both methods aim to improve the decision made related to the stowage strategy. The first approach is an iterative method that iteratively improves these decisions. In the second approach, a MIP model is solved to determine the stowage strategy decisions. The results show minor changes in the objective value, and there is still room for improvement.

In Chapter 3 a stochastic version of the cargo mix problem is considered. Here the aim is to include the unreliability with respect to the demand forecast in the industry by considering the cargo-flows as being stochastic. Instead of considering a circular route, a string of ports are considered, and the vessel is assumed to already be loaded with containers. We show that the problem cannot be solved using standard MIP solvers. This is mainly attributed to stowage strategy and the number of scenarios. To deal with this, we heuristically set the stowage strategy decisions, and a Rolling Horizon Heuristic is used to handle the number of scenarios. This matheuristic is shown to be scalable and finds high-quality solutions even for real-life instances. The contribution of this paper is three-fold. First, the problem definition is extended to include stochastic cargo flow. Second, the compact formulation is extended to include stochastic cargo flows. Lastly, a matheuristic is proposed to solve the problem. The work has been disseminated as follows:

- A paper co-authored with Alan Erera and Dario Pacino, soon to be submitted to Transportation Research Part E. (Christensen et al., 2017a)
- Presentation by Jonas Christensen at *7th International Conference on Logistics and Maritime Systems (LOGMS 2017)*. (Christensen et al., 2017b)

1.4.2 The Flexible Ship Loading Problem

Another consequence of the mega-container ships is on the terminal side. Bigger vessels require more crane moves per vessel, and terminals are under pressure to minimise the turnaround time for the vessels. Minimizing the turnaround time makes it possible for the carriers to realise more of the savings potential that comes with the bigger vessels, as they will not have to catch-up on the sea to stay on schedule because of port delays. For the terminal, improving productivity and minimising turnaround times helps to free up berth positions, and clears up capacity for another vessel. (JOC, 2014)

With regards to productivity, Asia's container terminal outranks the U.S. and European counterparts. This is mainly contributed to ports being operated 24 hours a day, a high degree of automation and a large transhipment volume. However, Subhangshu Dutt, executive director of Krishnapatnam port in Southeast India has expressed concerns maintaining this productivity level with bigger vessel becoming more common: "*Productivity will not increase, and even maintaining the level is a challenge. Longer vessels of 440 yards (such as Maersk Line's Triple E class) will lead to more berth wastage... It appears that productivity levels have plateaued, and unless there is a totally new innovative concept with regard to yard design, management and flows, there may not be any significant improvement*". The Flexible Ship Loading Problem (FSLP) investigates such an innovative concept using optimisation. (Mooney, 2015; JOC, 2014)

Acknowledging that improving terminal productivity is a shared goal between the carrier and the terminal, the FSLP investigates a collaboration between the terminal and liner shipping companies. The liner provides the terminal with a stowage plan based on container classes. The terminal then has the flexibility of determining the position of the specific containers, as long as it adheres to the provided stowage plan. The terminal will assign containers to specific slots on the vessel, while also scheduling transfer vehicles to retrieve the container from the yard and deliver it in front of the crane. Doing so the terminal can better plan what container to be loaded at what time, thus giving the terminal better conditions to minimise the turnaround time for the vessel. The terminal also benefits from this collaboration as they can plan the use of their container-handling equipment better, so it can be used elsewhere for another vessel.

The flexible ship loading problem is considered in Chapter 4 and Chapter 6. In Chapter 4 the problem is described more thoroughly, and the different components and interactions of the problem are described. The problem is formulated as a mathematical model, and a number of valid inequalities are tested. The model can only be used to solve small instances, and instead, a GRASP heuristic is used. The GRASP heuristic is shown to be scalable and can be used to find high-quality solutions in reasonable time. This paper contributes to the literature by first introducing the problem aimed at improving the productivity of the loading operations at container terminals. Secondly, the problem is formulated as a mathematical model, and a number of valid inequalities are described and tested. The model is shown to be intractable for large-scale

instances, and therefore a heuristic method is proposed. This work has been disseminated as follows

- A paper co-authored with Çagatay Iris, Dario Pacino, and Stefan Ropke submitted to Transportation Research Part B. (Iris et al., 2017a)
- Presentation by Dario Pacino at *OR2017 Berlin*. (Iris et al., 2017b)

Chapter 6 extends the work presented in Chapter 4. A new mathematical formulation is presented. The revised mathematical formulation is shown to provide better upper and lower bounds for the problem. Also, a hybrid heuristic based on the GRASP heuristic is presented. The hybrid heuristic finds better solutions compared to the GRASP and does so using less time. Chapter 6 also describes a column generation algorithm for the FSLP. The column generation approach seems promising but requires more work to properly evaluate. With additional work we believe the work presented in Chapter 6 can be the basis for one or more standalone journal papers.

1.5 Conclusions

In search of economies of scale container shipping lines are building bigger and bigger vessels. These new mega-vessels provide large unit cost savings compared to older and smaller vessels. However, a significant amount of the costs savings is attributed to the emergence of slow steaming. Comparing modern mega-vessels with modern smaller vessels the cost savings are significantly lower, and the benefits of the mega-vessels are thus set to diminish when older medium-sized vessels are decommissioned. (OECD/ITF, 2015)

While mega-vessels may cut unit costs for carriers, the total system costs are not reduced. Additional costs for ports, insurance companies and transport providers lead to higher total system costs as vessel sizes grow. Also, the general network structure leads to more transhipments and fewer direct services. These added costs to the logistics system may outweigh the benefits of the mega-vessels. Building even bigger vessels is therefore not a viable solution to deal with the diminishing benefits of the mega-vessels. Instead, the industry must improve operational efficiency. (UNCTAD, 2016)

One way to improve operational efficiency is the utilisation of data to make better plans. However, the global container transportation system has reached a size where humans alone cannot easily evaluate the best plan. This thesis aims to describe prototypes of advanced decision support tools to aid in this planning process.

The thesis contributes to the Operations Research (OR) literature by providing insight into how data and optimisation methods can be used to improve operational efficiency within the domain of liner shipping. The work has been disseminated in international peer-review journals and conferences. The contributions cover modelling, methodology and applications.

Two problems are studied, the Cargo Mix Problem and the Flexible Ship loading Problem. The Cargo Mix problem (Chapter 2, 3 and 5) aims at analysing the cargo composition needed

for a vessel to maximise its utilisation on a given service. In Chapter 2 and 5 the problem is considered as being deterministic, while stochasticity is include in Chapter 3. The applications of such models are mostly strategic and tactical (driving freight rates, and improving the vessel capacity model in fleet deployment and network design problems). However, their analytical strength is better suited at the operational level, e.g. in uptake management. The solution methods are based on mathematical models, and they are easily adapted to perform various what-if scenario analysis. To provide value as an analytical tool, it is important the solution methods be fast so multiple scenarios can be analysed in a few minutes. Therefore the focus has been to provide fast solutions. However, more accurate versions are also presented which can have value for strategic and tactical decisions. These models provide insight into how the loading of different types of containers impacts the utilisation. It can thus be seen as a first step toward a more dynamic pricing scheme. Such a pricing scheme is key for revenue management systems and intelligent booking systems.

The Flexible Ship Loading Problem (Chapter 4 and 6) studies a collaboration between the terminal and the liner company. The liner provides the terminal with a stowage plan based on container classes. The terminal utilises the flexibility within the class-based stowage plan and makes a detailed operative stowage plan. The terminal does so while considering load sequencing, transfer vehicle dispatching and scheduling. The objective is to minimise the penalty incurred by finishing the operation later than an expected finishing time and the cost of operating the transfer vehicles. The terminal clearly benefits from this, but so does the liner company, as the terminal can finish the loading earlier and the vessel spends less time at port. Multiple solutions methods are described ranging from exact MIP models to heuristic methods (GRASP and Hybrid heuristic) and two lower bound methods (Column generation and a MIP model). This integrated approach is compared to solving it in a hierarchical manner, where the terminal receives an operative stowage plan. The results show that significant cost savings can be achieved. For all test cases, the savings is between $35\% - 65\%$, and the average savings is 50%.

1.6 Future work

Liner shipping companies are sensitive towards their operating data and are often described as being conservative. Instead of advanced planning tools, companies usually prefer their traditional planning approaches. Therefore, research in the field is challenged due to the unavailability of data. However, recently companies within the domain have started to collaborate with academia. With the increasing complexities of the planning tasks, one can only assume the openness toward academia will develop further over time.

For the industry to take advantage of research collaborations, academics must provide business insights useful for practitioners. Such insights can either be input to policymakers or prototypes for operational planning tools.

Despite the advancements outlined in this thesis, there is still potential for further improving operational efficiency. Operational Alliances are a way to provide a weekly schedule, and today Vessel Sharing Agreements (VSA) are an important part of liner shipping. This aspect can easily be incorporated in network design problems, by reducing capacities and fixing edges. However, careful analysis of dual information of these edges can potentially put a value on the VSAs. This

may be very valuable during negotiations and give suggestions for which VSAs to pursue.

Further collaboration between liner shipping companies and port operators can potentially improve operating efficiency for both companies. Vessels are usually assigned a berthing time window long before calling the port. In practice, it is not uncommon vessels miss the allocated time. This can lead to congestion at the port and extra waiting time for the vessels. Also, real-time port updates might make vessels able to anticipate waiting times and therefore lower its speed, saving bunker.

For researchers, the availability of data is often a problem, and the existence of a benchmark set can be of great help to stimulate research within the field. The LinerLib (Brouer et al., 2014) is a benchmark set for the Liner Shipping Network Design Problem (LSNDP), and after making it available, the LSNDP has received more attention by researchers. Additionally, to be able to compare methods to each other the research community must agree on unifying problem definitions. The industry can benefit from this increased attention by having faster and better solution methods. Hopefully, more benchmarks will be published in the future, and more researchers will use the already existing benchmarks.

In the following, we outline possible future research directions related to the specific problems studied in this thesis.

1.6.1 Improved modelling

The methods described in this thesis can be extended to handle additional details. To cater for the operational applications of the cargo mix models, one could introduce indicators for the vessel turnaround time such as crane split. Dispersing the workload over multiple cranes will improve the crane split and minimise the vessel turnaround time. It is especially important that the block assignment allows for multiple cranes to unload containers simultaneously. Thus, crane split considerations can be added to the methods by including it in the model presented in Section 5.3.

We see multiple ways to improve the modelling of the Flexible Ship Loading Problem. The first extension is to optimise the loading sequence, i.e. the order in which the positions are loaded in. An easier extension is the addition of TV operating time windows, to symbolise the TV is needed elsewhere at the terminal. Another possible extension is to pool the Transfer Vehicles (TVs) for all Quay Cranes (QCs). If pooling is allowed, TV waiting time might be used to retrieve a container for another crane. In the column generation method described in Section 6.4.1 the subproblem finds a plan for a single vehicle, the pooling aspect can thus be included within the subproblem with only small changes to the master problem. Lastly, by considering unload operations together with loading operations, the effect of dual cycling can be studied.

1.6.2 Improved solution methods

There are multiple ways to extend or improve the methods presented in this thesis. For the cargo mix problem we saw in Chapter 2 and 3 that the block assignment calculated in Phase I

contributes to most of the final gap. Thus an idea for improving the results is to revise the way the block assignment is calculated. That is precisely the focus of Chapter 5. The model-based method did improve the results, but not substantially. However, operational objectives like crane split can be included in this method. To improve the block assignment for the stochastic version (Chapter 3) it can be made scenario dependent. Also for the stochastic version, the subproblem can be solved in parallel and thus speed up the overall method.

The next steps in solving the Flexible Ship Loading Problem should focus on calculating better lower bounds. The revised model (R-FSLP) in Chapter 6 did improve the lower bounds, but the relative difference between the solutions and the lower bounds are still high. It is unclear if this is due to poor upper bounds or poor lower bounds. For this reason, we looked into column generation. While the pooling of vehicle can be added in this method, the chosen decomposition does not yield better lower bounds. Instead, another way to decompose the problem is suggested in the conclusion of Chapter 6. The idea here is to generate *QC-plans* instead of *TV-plans*. Doing so, the only constraints in the master problem concerns the number of QC-plans used, container resource constraints and the calculation of the end time. The calculation of the end time is a simple constraint as the QC end time is known from the QC-plans. The pricing problem will be similar to a 1-QC FSLP problem, but with no delay cost. Instead of the delay cost, there is a cost depending proportionally on the QC end time. We are currently looking into this, and preliminary results are promising.

References

Alphaliner (2017). *Alphaliner TOP 100*. Accessed: 26-09-2017. URL: `https://alphaliner.axsmarine.com/PublicTop100/`.

Ambrosino, D. and A. Sciomachen (1998). "A Constraint Satisfaction Approach for Master Bay Plans". In: *Water studies series* 36.

Ambrosino, D., D. Anghinolfi, M. Paolucci, and A. Sciomachen (2009). "A new three-step heuristic for the master bay plan problem". In: *Maritime Econ Logistics* 11.1, pp. 98–120. ISSN: 1479-2931.

Ambrosino, D., A. Sciomachen, and E. Tanfani (2004). "Stowing a containership: the master bay plan problem". In: *Transportation Research Part A: Policy and Practice* 38.2, pp. 81–99. ISSN: 09658564.

Aslidis, A. H. (1984). *Optimal container loading*. Master Thesis.

Australian Broadcast Company (2014). *How container shipping changed the world through globalisation*. URL: `http://www.abc.net.au/radionational/programs/rearvision/the-big-metal-box-that-changed-the-world/5684586`.

Avriel, M., M. Penn, N. Shpirer, and S. Witteboon (1998). "Stowage planning for container ships to reduce the number of shifts". In: *Annals of Operations Research* 76.1-4, pp. 55–71.

BBC (2017). *The simple steel box that transformed global trade*. URL: `http://www.bbc.com/news/business-38305512`.

Berglund, P. (2017). *Mega-Ships: Is Bigger Always Better?* Blog. URL: `https://www.xeneta.com/blog/mega-ships`.

Botter, R. C. and M. A. Brinati (1992). "Stowage Container Planning: A Model for Getting an Optimal Solution". In: *Proceedings of the IFIP TC5/WG5.6 Seventh International Conference on Computer Applications in the Automation of Shipyard Operation and Ship Design, VII*.

Amsterdam, The Netherlands, The Netherlands: North-Holland Publishing Co., pp. 217–229. ISBN: 0-444-89728-3.

Brouer, B. D., J. F. Alvarez, C. E. M. Plum, D. Pisinger, and M. M. Sigurd (2014). "A Base Integer Programming Model and Benchmark Suite for Liner-Shipping Network Design". In: *Transportation Science* 48, pp. 281–312. ISSN: 0041-1655.

Brouer, B. D., C. V. Karsten, and D. Pisinger (2017). "Optimization in liner shipping". In: *4OR* 15.1, pp. 1–35.

Christensen, J., A. Erera, and D. Pacino (2017a). "Modelling and solving the Stochastic Cargo Mix Problem". Soon to be submitted to Transportation Research Part E: Logistics and Transportation Review.

Christensen, J., A. Erera, and D. Pacino (2017b). "The Stochastic Cargo Mix Problem". Presented at 7th International Conference on Logistics and Maritime Systems (LOGMS 2017).

Christensen, J. and D. Pacino (2016). "A Matheuristic for the Cargo Mix Problem with Block Stowage". Presented at 9th Triennial Symposium on Transportation Analysis (TRISTAN 2016).

Christensen, J. and D. Pacino (2017). "A matheuristic for the Cargo Mix Problem with Block Stowage". In: *Transportation Research Part E: Logistics and Transportation Review* 97, pp. 151–171. ISSN: 1366-5545.

Christensen, J., D. Pacino, J. Fonseca, and H. Psaraftis (2015a). "Cargo Mix optimization in Liner Shipping". Presented at the hEART 2015 conference. URL: `www.heart2015.transport.dtu.dk/-/media/Sites/hEART2015/abstracts%20hEART/hEART_2015_submission_54.ashx?la=da`.

Christensen, J., D. Pacino, H. Psaraftis, and J. Fonseca (2015b). "Cargo-mix optimization in Liner Shipping". Presented at Sixth International Workshop on Freight Transportation and Logistics (Odysseus 2015). URL: `http://homepages.laas.fr/njozefow/odysseus2015.pdf`.

Christiansen, M., K. Fagerholt, B. Nygreen, and D. Ronen (2007). "Chapter 4 Maritime Transportation". In: *Transportation*. Ed. by C. Barnhart and G. Laporte. Vol. 14. Handbooks in Operations Research and Management Science Supplement C. Elsevier, pp. 189–284.

Christiansen, M., K. Fagerholt, B. Nygreen, and D. Ronen (2013). "Ship routing and scheduling in the new millennium". In: *European Journal of Operational Research* 228.3, pp. 467–483. ISSN: 0377-2217.

Christiansen, M., K. Fagerholt, and D. Ronen (2004). "Ship Routing and Scheduling: Status and Perspectives". In: *Transportation Science* 38.1, pp. 1–18.

Delgado, A. (2013). "Models and Algorithms for Container Vessel Stowage Optimization".

Delgado, A., R. Jensen, and K. Janstrup (2012). "A constraint programming model for fast optimal stowage of container vessel bays". In: *European Journal of Operational Research* 220.1, pp. 251–261.

Ding, D. and M. C. Chou (2015). "Stowage Planning for Container Ships: A Heuristic Algorithm to Reduce the Number of Shifts". In: *European Journal of Operational Research*. ISSN: 03772217.

Dubrovsky, O., G. Levitin, and M. Penn (2002). "A genetic algorithm with a compact solution encoding for the container ship stowage problem". In: *Journal of Heuristics*, pp. 585–599.

Dupzyk, K. (2015). *The Two Greatest Canals in the World Are Getting Even Bigger*. URL: `http://www.popularmechanics.com/technology/infrastructure/a16725/panama-suez-canal-projects/`.

Hapag-Lloyd (2017). *Container Specification*. URL: `https://www.hapag-lloyd.com/content/dam/website/downloads/pdf/17038_Update_Container_Specification_engl_sRGB.pdf`.

Iris, Ç., J. Christensen, D. Pacino, and S. Ropke (2017a). "Flexible ship loading problem with transfer vehicle assignment and scheduling". Submitted to Transportation Research Part B: Methodological.

Iris, Ç., J. Christensen, D. Pacino, and S. Ropke (2017b). "The Flexible Ship Loading Problem". Presented at OR2017 in Berlin. URL: http://www.or2017.de/general-information/index.html.

Iris, C., A. Larsen, S. Ropke, and D. Pacino (2016). "Exact and Heuristic Methods for Integrated Container Terminal Problems". PhD thesis.

JOC (2014). "Berth Productivity: The Trends, Outlook and Market Forces Impacting Ship Turnaround Times". In: *JOC Port Productivity*. JOC Group. URL: https://www.joc.com/whitepaper/berth-productivity-trends-outlook-and-market-forces-impacting-ship-turnaround-times-0.

Kang, J.-G. and Y.-D. Kim (2002). "Stowage planning in maritime container transportation". In: *Journal of the Operational Research Society* 53.4, pp. 415–426.

Kim, K. H., J. S. Kang, and K. R. Ryu (2004). "A beam search algorithm for the load sequencing of outbound containers in port container terminals". In: *OR Spectrum* 26.1, pp. 93–116.

Li, F., C. Tian, R. Cao, and W. Ding (2008). "An integer linear programming for container stowage problem". In: *Computational Science–ICCS 2008*, pp. 853–862.

Low, M., X. Xiao, F. Liu, S. Huang, W. J. Hsu, and Z. Li (2009). "An Automated Stowage Planning System for Large Containerships". In: *Proceedings of the 4th Virtual Int. Conference on Intelligent Production Machines and Systems*.

Maersk Line (2016). *Welcome to the future Maersk app store*. URL: https://www.maersk.com/en/stories/welcome-to-the-future-maersk-app-store.

Meisel, F. (2009). *Seaside Operations Planning in Container Terminals*. Physica-Verlag Heidelberg.

Meng, Q., S. Wang, H. Andersson, and K. Thun (2014). "Containership Routing and Scheduling in Liner Shipping: Overview and Future Research Directions". In: *Transportation Science* 48.2, pp. 265–280.

Mooney, T. (2015). *Lagging berth productivity costs shipping billions*. Blog. URL: https://www.joc.com/port-news/port-productivity/lagging-berth-productivity-costs-shipping-billions_20151218.html.

Morley, H. R. (2016). *Container shipping overcapacity forecast to worsen*. URL: https://www.joc.com/maritime-news/container-lines/container-shipping-overcapacity-forecast-worsen_20161102.html.

OECD/ITF (2015). "The Impact of Mega-Ships". In: *Case-Specific Policy Analysis*. OECD/ITF. URL: https://www.itf-oecd.org/impact-mega-ships.

OOCL (2017). "OOCL reaches milestone with the christening of the OOCL Hong Kong". In: *OOCL reaches milestone with the christening of the OOCL Hong Kong*. URL: http://www.oocl.com/eng/pressandmedia/pressreleases/2017/Pages/12may17.aspx.

Pacino, D. (2012). "Fast Generation of Container Vessel Stowage Plans: using mixed integer programming for optimal master planning and constraint based local search for slot planning". PhD Thesis. PhD thesis.

Pacino, D., A. Delgado, R. M. Jensen, and T. Bebbington (2011). "Fast Generation of Near-Optimal Plans for Eco-Efficient Stowage of Large Container Vessels". In: *Computational Logistics*. Ed. by J. W. Böse, H. Hu, C. Jahn, X. Shi, R. Stahlbock, and S. Voß. Vol. 6971. Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 286–301.

Pacino, D. and R. Jensen (2012). "Constraint-based local search for container stowage slot planning". In: *Lecture Notes in Engineering and Computer Science* 2, pp. 1467–1472. ISSN: 2078-0958.

Parreno, F., D. Pacino, and R. Alvarez-Valdes (2016). "A GRASP algorithm for the container stowage slot planning problem". In: *Transportation Research Part E: Logistics and Transportation Review* 94, pp. 141–157. ISSN: 1366-5545.

PCB (2017). *Malcolm McLean – A Salute to the Brilliance of Carriers.* Blog. URL: https://blog.pcb.ca/2017/02/malcolm-mclean-salute-brilliance-carriers-02-2017/9137.

Ronen, D. (1983). "Cargo ships routing and scheduling: Survey of models and problems". In: *European Journal of Operational Research* 12.2, pp. 119–126. ISSN: 0377-2217.

Ronen, D. (1993). "Ship scheduling: The last decade". In: *European Journal of Operational Research* 71.3, pp. 325–333. ISSN: 0377-2217.

Sciomachen, A. and E. Tanfani (2003). "The master bay plan problem: a solution method based on its connection to the three-dimensional bin packing problem". In: *IMA Journal of Management Mathematics* 14, pp. 251–269.

Shipmap.org (2016). *Shipmap.org - Visualisation of Global Cargo Ships.* Research by University College London Energy Institute, visualisation by Kiln. URL: https://www.shipmap.org/.

Steenken, D., S. Voss, and R. Stahlbock (2004). "Container terminal operation and operations research - A classification and literature review". In: 26, pp. 3–49.

Stopford, M. (2009). *Maritime economics 3e.* Routledge.

The Economist (2001). *Malcolm McLean.* URL: http://www.economist.com/node/638561.

The Economist (2013a). *The humble hero.* URL: https://www.economist.com/news/finance-and-economics/21578041-containers-have-been-more-important-globalisation-freer-trade-humble.

The Economist (2013b). *Why have containers boosted trade so much?* Blog. URL: https://www.economist.com/blogs/economist-explains/2013/05/economist-explains-14.

UNCTAD (2016). "Review of Maritime Transport 2016". In: *United Nations Conference on Trade and Development.*

Velta (2016). *The importance of the worlds canals to global trade.* URL: http://www.velta.co.uk/news/229/76/The-importance-of-the-worlds-canals-to-global-trade/.

Wilson, I. and P. Roach (2000). "Container Stowage Planning: A Methodology for Generating Computerised Solutions". In: *The Journal of the Operational Research Society* 51.11, pp. 1248–1255.

World Shipping Council (2013). *History of Containerization: Industry Globalization.* URL: http://www.worldshipping.org/about-the-industry/history-of-containerization/industry-globalization.

Part I

Research papers

Chapter 2

A Matheuristic for the Cargo Mix Problem with Block Stowage

Jonas M. Christensen[a] · Dario Pacino[a]

[a]Management Science, Department of Management Engineering, Technical University of Denmark, Produktionstorvet, Building 424, DK-2800 Kgs. Lyngby, Denmark

Publication Status: Published as *J. Christensen and D. Pacino (2017). "A matheuristic for the Cargo Mix Problem with Block Stowage". In:* Transportation Research Part E: Logistics and Transportation Review *97, pp. 151–171.* ISSN*: 1366-5545*

Abstract: The cargo-mix problem aims at selecting the amount of containers of a given type to load on a vessel. In this article we present an extended definition that includes the analysis of a circular route with draft restrictions, limitations on expected cargo and the use of a block stowage strategy. A compact formulation of the problem based on the state-of-the-art heuristic decomposition is shown not to be able to solve the extended problem, thus a matheuristic approach is presented that can achieve high quality results in a matter of seconds.

Keywords: Liner shipping · Cargo composition · Maritime logistic · Block stowage · Stowage planning · Matheuristic

2.1 Introduction

Aside from the few years of financial crisis, the liner shipping industry has had a continuous growth. The demand for efficient and cheap transportation and a fierce competition has driven shipping rates down, making it paramount for the carriers to utilise their vessels as efficiently as possible.

Focus on vessel intake maximisation is no news for the shipping lines. Container vessels are delivered with a nominal capacity that ship owners know is only theoretical. Unless the cargo weight distribution is perfect, the nominal intake cannot be reached. Stowage coordinators fight this battle every day. They are the planners of the cargo and have to find a load configuration (*stowage plan*) that both suits the current cargo to be loaded but also guarantees that the vessel can be utilised to its maximum in future ports. The size of nowadays vessels is, however, making this work harder (Pacino et al., 2011). Moreover, the cargo composition available in the different ports might not be suitable for the full utilisation of the vessel. To give a brief example, consider a vessel that has to load a high number of very heavy containers. As a consequence, the *draft* of the vessel (the distance between the waterline and the bottom of the hull) will be greater. If the vessel is scheduled to visit a port with a low water depth, stowage coordinators will have to leave a number of containers behind to reduce the draft and avoid the risk of running the vessel aground. Leaving containers behind is clearly not a desirable situation.

The focus of our work is the analysis of vessels' cargo mix (*the cargo mix problem*), in particular finding what cargo composition is needed for a vessel to maximise its utilisation on a given service. Differing from stowage planning, where a list of pre-selected containers must be stowed on the vessel, the cargo mix problem aims at selecting the quantity of containers of each type that should be loaded on a vessel to maximise its intake. Moreover, standard stowage planning approaches only consider the current port, whereas the cargo mix problem considers a cyclic service with multiple ports. The proposed model can have multiple applications ranging from driving rates prices, improving fleet composition and network design (Christiansen et al., 2007; Reinhardt and Pisinger, 2012; Brouer et al., 2014), or analyse the difference between an expected cargo load and an optimal one.

In the work of Delgado (2013), it was shown that a cargo-mix analysis based on simple capacity limitations greatly overestimates the revenue of the vessel. The presented model, however, did not include a number of features which we deem essential for a correct capacity or revenue calculation.

In our work, we propose an extension of the cargo-mix problem that includes draft restrictions at the visited ports and a block stowage strategy. This last refers to the logical partitioning of the vessel into blocks. Each block is then allowed to host only containers that have the same discharge port. This alternative way of stowing containers is aimed at improving operations at ports, since it makes it possible to perform e.g. dual cycling (where load and discharge operations are no longer sequential). Examples of block stowage are found in the stowage planning literature in e.g. Ambrosino et al. (2015a).

Our work contributes to the state-of-the-art in the following way. First we extend the formal definition of the cargo mix problem from Delgado (2013) to include block stowage, circular routes,

draft restrictions, and cargo flow limitations. Second, we propose an extension of the compact formulation of Delgado (2013) where the new problem features are included. We also show that the new model can no longer be solved via mixed-integer programming. At last, we propose and compare a number of matheuristics that are able to solve the problem in a matter of seconds.

The remainder of the paper is organised as follows. Section 2.2 presents the necessary background knowledge. In Section 2.3, the existing relevant literature is reviewed. Section 2.4 describes the problem in deeper details and presents a compact formulation for the problem. Section 2.5 describes the matheuristic approach. Section 2.6 is a brief explanation of the data used, and Section 2.7 presents the results for the MIP model as well as the matheuristic. Lastly Section 2.8 contains final remarks and conclusions.

2.2 Background

Optimising the intake of a vessel requires deep understanding of vessel architecture, and the industry as a whole. Liner shipping companies transport containers between ports on a fixed cyclic schedule. Most containers transported on liner vessels are 8 feet wide, 8'6" high and *20'*, *40'* or *45'* long. *High-cube* containers also exist, which are 9'6" high. Some containers have an integrated cooling unit for the transport of e.g. perishable goods. These containers are called *reefers* and must be plugged into an electrical outlet, and thus can only be stowed where such a plug is available. Additionally, there are dangerous goods containers (*IMO containers*) which must follow strict segregation rules.

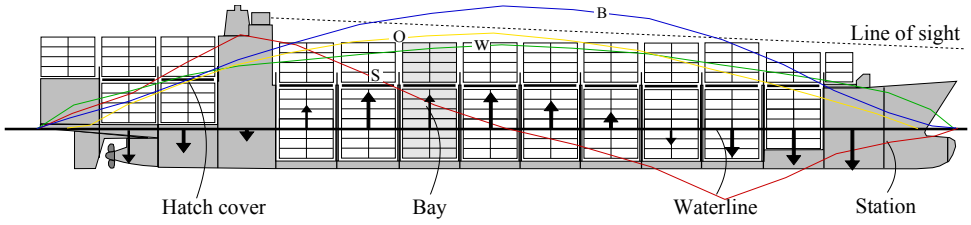


Figure 2.1: Example figure of a container vessel.

Figure 2.1 shows the layout of a container vessel. A vessel is divided into *bays*. Each bay is divided into an *on-deck* and *below-deck* part, physically separated by a *hatch cover*. Hatch covers are flat leak proof structures that can be removed when loading/unloading containers. Each bay consists of a number of *stacks* divided into *slots*. A slot can hold a single 20' container. Figure 2.2a shows a transversal section of a bay, where the numbers denote the blocks, and Figure 2.2b shows the design of a stack within a bay. Two slots side by side are denoted a *cell*, each consisting of an *aft* (towards the stern) and a *fore* (towards the bow) slot. Each stack has weight and height limits. A stack has outer and inner supports and, as shown in Figure 2.2b, a 20' container uses the inner and the outer supports, whereas a 40' container is supported only by the outer supports. A stack thus has two weight limits, one for the outer supports and one for the inner supports. Due to the construction of the vessel the inner supports are weaker than the outer supports. Twist locks can

(a) A bay in the vessel *(b) The layout of a stack*

Figure 2.2: Example figure showing the design of a bay, and a stack.

be used to lock containers to each other at the corners. There are no such locking mechanisms in the middle of 40' containers, hence stacking rules impose that 20' containers cannot be stowed above 40' containers. Twist locks have limited strength, thus container weights must normally decrease upwards in stacks on deck. Below deck, cell guides secure containers in place.

Vessels capacity is measured in *Twenty-foot Equivalent Units* (TEU), a 40' standard and 40' high-cube container are 2 TEU, or 1 *FEU* (*Forty-foot Equivalent Unit*). In liner shipping profit comes from the transported containers, and naturally, carriers want to utilise the full capacity to increase revenue. The main expenses are fuel cost and costs associated with port stays. Reducing the duration of a port stay can be done by e.g. reducing *overstowage*. Overstowage happens when a container is stowed below a container destined for a later port. Containers can overstow in two different ways. *Stack overstowage* occurs when the overstowing container is in the same stack as the overstowed container and *hatch overstowage* occurs when a container on deck is overstowing a container below deck. If hatch overstowage occurs all containers above the hatch cover must be restowed, in order to unload the overstowed container.

A current trend that is becoming popular in the industry, is the notion of *block stowage*. To facilitate better planning and ease port handling operations, the ship is divided into blocks (a logical grouping of stacks). Containers in a block must, as a rule of thumb, have the same discharge port. There can be different definitions of a block, often company dependent. Here, according to our industrial collaborator, stacks below and above the same hatch-cover belong to the same block. This means that, given that all containers within the block must have the same discharge port, overstowage and hatch-overstowage cannot happen. Moreover, as it can also be seen in Figure 2.2a, external blocks are seen as one to ease transverse stability (as modelled in e.g. Pacino et al. (2011)).

Any stowage plan must result in a seaworthy vessel. A vessel is declared seaworthy if its static stability is correct and all stress forces are within limits. Stress limits are only known at specific points across the length of the vessel, these calculation points are called *frames*. Figure 2.1 shows a longitudinal section of a vessel with 14 frames. The W curve shows the weight distribution across the length of the vessel, and the O curve represents the buoyancy force. The arrows show the resulting force acting on each of the sections. If these sections were allowed to move freely as independent objects, some of them would sink deeper into the water and some would rise depending on the resulting force acting on the section. However, they are not allowed to move

freely, and thus this causes shearing and bending stresses over the vessel structure. The shearing and bending limits are known for every frame. These limits must be satisfied when leaving each port. Shear force is shown in Figure 2.1 as the curve S, and the bending moment is the curve B.

Most container vessels are equipped with *ballast tanks*. These tanks can be used to load water in order to achieve a better weight distribution, reduce stress forces, or fix stability issues. However, the use of these tanks is discouraged as this would increase the total weight (*displacement*) of the vessel, thus increasing fuel consumption.

Additional stability limits must be held for the ship to be declared seaworthy, e.g. *trim*. The *aft draft* is the distance between the waterline and the bottom of the hull at the stern of the ship, and the *fore draft* is at the bow. Trim is defined as the difference between the aft draft and the fore draft, and must be within its limits. Moreover, the height a stack can reach is limited by the *line of sight*. The captain standing on the bridge must be able to see a distance equal to the double of the length of the vessel, or 500 meters, whichever is smallest. The line of sight depends on the draft of the vessel and is thus not simply a stack capacity reduction. The *metacentric height* describes the vessel's ability to return to its initial position when affected by an external force. Figure 2.3 shows the transversal section of a vessel. The metacentric height is defined as the distance between the point M (*metacentre*) and the point G (the centre of gravity). The metacentric height must be within operational limits to ensure that the vessel will not capsize.

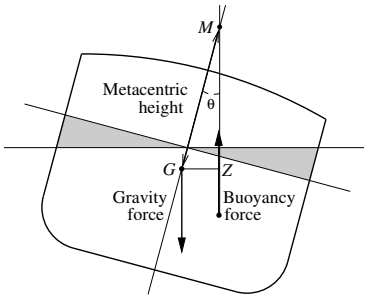


Figure 2.3: Metacentric height of a vessel

If the described stacking rules, stability, and stress limits are held at bay then the vessel can be declared seaworthy, and can hereafter safely depart.

A vessel is fully described by a document called *vessel profile*, which specifies all the weight and capacity limits, a general outline of the vessel etc. The non-linear functions for the draft, centre of gravity, trim and metacentre are outlined in the *hydrostatic table*, within the vessel profile. We refer the reader to Delgado (2013) for a more detailed description of the stability requirements and stacking rules.

2.3 Literature Review

Academic work on the liner shipping cargo mix problem is limited. The first formal description was presented in the PhD thesis of Delgado (2013) where an integer programming model was

presented. The author shows that for multi-port scenarios the model does not scale. To achieve scalability, the same decomposition as in earlier stowage planning work (Pacino et al., 2011) was applied. The results show that the decomposition gives an accurate estimate. Tested on 360 instances, on a service of up to 10 ports, the approach was able to optimally solve 91.7% of the instances, with a time limit of 60 minutes. In comparison to ours, the work in Delgado (2013) is only able to analyse a portion of a service and not a full rotation. Moreover, there is no restriction to the number of containers that can be selected, whereas our version of the problem is bounded to the expected cargo-flows of the analysed service, and imposes draft restrictions on the visited ports. Our work also shows that the inclusion of a block stowage strategy makes the MIP model proposed in Delgado (2013) intractable for industrial size instances.

In Zurheide and Fischer (2011) the authors presents a slot allocation model which helps the carrier decide whether to accept or reject a booking request. The decision takes into account the possibility to postpone a container in order to accept a more beneficial one. A revenue model for a short-sea shipping service is also presented in Feng and Chang (2008). The model disregards major aspects of stowage planning and only consider TEU and weight capacity. In Delgado (2013), it was shown that this model due to the inaccuracies, in average, overestimates the revenue by 8%. In this work, we fill the gaps in the literature by describing a fast and scalable heuristic that considers cargo flows, circular routes and draft at port. Inspired by previous work we solve a master problem based on stowage planning as Delgado (2013) has shown this can be used to accurately approximate the revenue. In the future, we envision to extend this work by considering stochastic cargo flows, and the extension of the analysis to multiple vessels.

Given the limited amount of academic work on liner shipping cargo-mix problems, we find it relevant to introduce the literature related to stowage planning since the Cargo Mix Problem with Block Stowage (CMPBS) could be seen as a generalisation. The main difference between the CMPBS and stowage planning is that CMPBS performs cargo selection based on a set of expected cargo flows between the visited ports, whereas in stowage planning a list of containers to load has already been selected. Stowage planning is mainly concerned with the loading of the containers at the current port; in contrast, CMPBS analyses a cyclic route. In the work of Pacino et al. (2012) it was shown that vessel stability constraints are non-linear if the cargo weight is variable, which is the case in the CMPBS. These non-linearities, however, can be approximated with a small number of linear planes without losing too much information, as described in Delgado (2013). Botter and Brinati (1992) presents an accurate formulation which includes metacentric height, transversal stability, trim, shear forces, and bending moment as constraints. The model fails to solve for small vessels with 1000 TEU capacity, and therefore a heuristic decomposition is proposed. Integer programming models assigning containers to slots are also presented in Ambrosino et al. (2004) and Li et al. (2008). The main stability consideration enforced in these models is that the left and right weight difference and bow and stern weight difference must be within a certain predefined limit. However, scalability issues of these models make them unsuitable for practical use. Integer programming models for stowage planning, assigning containers to specific slots, experience scalability issues due to the large number of variables and constraints. Scalable approaches are based on multi-phase methods. Both Kang and Kim (2002) and Wilson and Roach (2000) describes a 2-phased approach, where the former is of an iterative nature, and in the latter the flow of information only goes downward. A three-step heuristic is introduced in Ambrosino et al. (2009), in which the left and right, and bow and stern weight differences are kept within a tolerance. Pacino et al. (2011) presents a solution method which takes most of the stability issues into consideration. The approach is a 2-phased method where

first containers are distributed to subsections of the vessel ensuring that the metacentric height, draft, trim and shear forces are within limits. In the second phase containers are then assigned to specific position. The most accurate published model is, however, the one described in Pacino et al. (2012) which, along with the stability and stress limits considered in Pacino et al. (2011), also includes direct linearizations of hydrostatic data and the modelling of ballast tanks.

Ambrosino et al. (2010) compares an ant colony heuristic and a tabu search heuristic, and Avriel et al. (1998) presents a model that minimise the number of shifts (similar to overstowage) in stowage planning. However, the model has limited applicability due to scalability issues, and therefore the authors developed the Suspensory Heuristic. The Suspensory Heuristic only focuses on container stacking constraints and does not include any stability considerations. Dubrovsky et al. (2002) presents a genetic algorithm (GA) for the minimum shifts problem. The GA obtain results similar to those obtained by the Suspensory Heuristic but has the possibility of incorporating additional constraints, such as simple stability constraints. Ding and Chou (2015) considers the same problem and describes a state-of-the-art heuristic that improves the results from the Suspensory Heuristic. Sciomachen and Tanfani (2003) uses a 3D-bin packing heuristic to tackle the master bay planning problem. Constraint programming approaches worth mentioning for stowage planning includes Ambrosino and Sciomachen (1998), Delgado et al. (2012) and the constraint-based local search method in Pacino and Jensen (2012).

Of interest is also the work of Ambrosino et al. (2015b) and Ambrosino et al. (2015a) that tackles the Multi Port Master Bay Planning Problem (MP-MBPP), where the authors include block stowage constraints similar to those of the CMPBS. The vessel is divided into sections according to their hatch covers, and containers in the same section must have the same destination. In the MP-MBPP all containers must be loaded, and the objective minimises a weighted sum of the crane makespan and number of hatch-overstowing containers. In Ambrosino et al. (2015b) the problem is thoroughly described, and formulated as a mixed integer program. In Ambrosino et al. (2015a) a heuristic based on fixing variables in the MIP model is presented, and it is shown to outperform the MIP model for the 16 tested instances. The method is tested on ships of varying size and using a service of 6 ports. The MP-MBPP considers a sequence of ports, and not a cyclic route as in the CMPBS. Furthermore, the MP-MBPP uses a fixed load list whereas an expected cargo flow is used as input to the CMPBS. Also, the CMPBS imposes a higher degree of accuracy wrt. the stability of the vessel. These differences in the two problems make the heuristic in Ambrosino et al. (2015a) unsuitable for the CMPBS.

2.4 The Cargo Mix Problem

Given a liner service, a vessel profile and the expected cargo flows, the liner shipping CMPBS aims at finding a mixture of container types to ideally load on the vessel at each port of call. The cargo mix must fulfil the stability requirements and respect the expected cargo flows while optimising a function of cargo values, which in this work is represented by the intake and the revenue. The block stowage requirement is strictly enforced in the CMPBS, thus the containers in each block must have the same discharge port.

The problem is decomposed similarly to earlier work (Kang and Kim, 2002; Wilson and Roach, 2000; Ambrosino et al., 2010; Pacino et al., 2011; Delgado, 2013), instead of assigning specific

containers to a specific slot, containers are divided into *container types*, and container types are assigned to blocks. Each container type represents a number of containers with the same properties in terms of weight, height, length and reefer capabilities. The decomposition is illustrated in Figure 2.4. In the master planning phase, a number of containers from a container type are assigned to a block while satisfying high-level capacity constraints. The slot planning phase considers each block independently and assigns a specific slot for each of the containers assigned in the master planning phase. In this work, we disregard the slot planning phase since, as Delgado (2013) have shown, the master planning phase gives an accurate estimation.

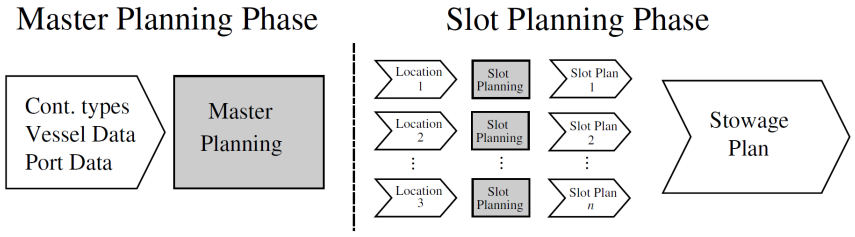


Figure 2.4: The two-phase heuristic decomposition

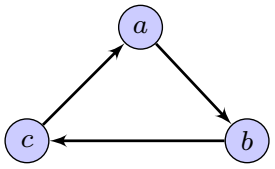
The objective of the CMPBS is not to generate fully feasible stowage plans, but to evaluate the capacity and utilisation of a vessel, given a service. We consider the most standard containers: 20' dry and reefer containers, 40' dry and reefer containers both normal height and high-cube. The vessel is enforced to be at even keel (with a trim of zero), and all the previously described stability and stress limits must be held at bay.

The Cargo Mix Problem with Block Stowage can be modelled as a mixed integer program. First, define $\mathcal{C}$ as the set of container types. A container type, $c \in \mathcal{C}$, is defined by the dimensions and properties of the containers and the parameters Γ^c, Φ^c, v^c describes respectively the TEU coefficient (1 for a 20' container, 2 for a 40' container), volume and weight of container type $c \in \mathcal{C}$. The volume is needed for the modelling of the high-cube containers, which need further restrictions than a simple capacity constraint. Also define the set $\mathcal{R} \subset \mathcal{C}$, as the set of all reefer container types, and let the set $\mathcal{B}$ be the set of blocks. In the CMPBS a full cyclic service is considered and $\mathcal{P}$ is the set of ports visited in one rotation. The container demand is described by the set $\mathcal{T}$ and the parameter a^{tc}. The set $\mathcal{T}$ is the set of transports. A transport describes an origin-destination pair and a^{tc} is the number of containers available of type $c \in \mathcal{C}$ during transport $t \in \mathcal{T}$. The set $\mathcal{T}_p^{ON}$ is the set of transports where port p is visited between the origin and destination of the transport. Define P_i^j as the ports visited between i and j, including i but excluding j, then $\mathcal{T}_p^{ON}$ is defined as follows

$$\mathcal{T}_p^{ON} = \{t \in \mathcal{T} \,|\, p \in P_{o^t}^{d^t}\} \qquad \forall p \in \mathcal{P}$$

Where o^t and d^t is the origin and destination port of transport t. Thus for a given port p the set $\mathcal{T}_p^{ON}$ consists of those transports that visit port p on the route from its origin to destination. As an example, Figure 2.5 shows a graph of a vessel rotation, the possible transports and the transports that are in the sets $\mathcal{T}_a^{ON}$, $\mathcal{T}_b^{ON}$, $\mathcal{T}_c^{ON}$.

The main decision variable is $y_b^{tc} \in \mathbb{N}$ defining the number of containers of container type $c \in \mathcal{C}$ of transport $t \in \mathcal{T}$ to be stowed in block $b \in \mathcal{B}$. Furthermore, to ensure that the containers in a block have the same destination port, at each port we need to assign a destination for each



t	o^t	d^t
1	a	b
2	a	c
3	b	c
4	b	a
5	c	a
6	c	b

$$\mathcal{T}_a^{ON} = \{1, 2, 6\}$$
$$\mathcal{T}_b^{ON} = \{3, 4, 2\}$$
$$\mathcal{T}_c^{ON} = \{5, 6, 4\}$$

Figure 2.5: Example illustration of the sets $\mathcal{T}_p^{ON}$

block. This decision is captured in the binary variable σ_{bp}^d, defined as follows

$$\sigma_{bp}^d = \begin{cases} 1, & \text{If block } b \in \mathcal{B} \text{ is assigned destination } d \in \mathcal{P} \text{ at port } p \in \mathcal{P} \\ 0, & \text{Otherwise.} \end{cases}$$

Lastly, let $w_{bp} \in \mathbb{R}^+$ denote the total weight stowed in block $b \in \mathcal{B}$ at port $p \in \mathcal{P}$.

The main contribution of this paper is not the modelling of the stability constraints. Hence, to simplify the forthcoming model, we define the polyhedron $\mathcal{W}$ as the weight allocations that results in a stable vessel.

$$\mathcal{W} = \left\{ \mathbf{w} \in \mathbb{R}^{|\mathcal{B}|\cdot|\mathcal{P}|} \;\; s.t. \;\; (2.24) - (2.43) \right\}$$

In essence, the polyhedron $\mathcal{W}$, is a set of constraints ensuring the seaworthiness of the vessel. The constraints are described in Section 2.A, and are based on the work by Delgado (2013) which also gives a more thorough description. This simplifies the forthcoming model, and keeps the focus on our contributions, that is, the addition of the block stowage requirement, circular routes and cargo flow limitations.

Below, all the sets, variables, and parameters are summarised, and additional sets and parameters are introduced.

Sets:

$\mathcal{P}$	Set of ports
$\mathcal{T}$	Set of transports
$\mathcal{T}_p^{ON}$	Set of transports that visits port $p \in \mathcal{P}$
$\mathcal{C}$	Set of container types
$\mathcal{C}^{20} \subset \mathcal{C}$	Set of container types with length 20 feet.
$\mathcal{R} \subset \mathcal{C}$	Set of reefer container types
$\mathcal{R}^{20} \subset \mathcal{R}$	Set of reefer container types, with length 20 feet
$\mathcal{B}$	Set of blocks.

Variables:

$y_b^{tc} \in \mathbb{N}$	Number of containers of type $c \in \mathcal{C}$ of transport $t \in \mathcal{T}$ to be stowed in block $b \in \mathcal{B}$.
$w_{bp} \in \mathbb{R}^+$	Weight of the containers stowed in block $b \in \mathcal{B}$ at port $p \in \mathcal{P}$.
$\sigma_{bp}^d \in \{0,1\}$	Variable denoting whether or not block $b \in \mathcal{B}$ can only contain containers with destination $d \in \mathcal{P}$ at port $p \in \mathcal{P}$.

Parameters:

$f^{tc} \in \mathbb{R}^+$	The value of container type $c \in \mathcal{C}$ of transport $t \in \mathcal{T}$.
$u^{tc} \in \mathbb{R}^+$	Revenue of container type $c \in \mathcal{C}$ of transport $t \in \mathcal{T}$.
$p^t \in \mathbb{N}$	Number of ports visited by transport $t \in \mathcal{T}$.
$d^t \in \mathcal{P}$	Destination port for transport $t \in \mathcal{T}$.
$k_b^{TEU} \in \mathbb{N}$	Teu capacity for block $b \in \mathcal{B}$
$k_b^{20} \in \mathbb{N}$	Capacity for 20' containers for block $b \in \mathcal{B}$
$r_b^{Cell} \in \mathbb{N}$	Reefer cell capacity of block $b \in \mathcal{B}$
$r_b^{Slot} \in \mathbb{N}$	Reefer slot capacity of block $b \in \mathcal{B}$
$\Gamma^c \in \{1,2\}$	TEU coefficient of container type $c \in \mathcal{C}$
$h_b \in \mathbb{R}^+$	Volume capacity for block $b \in \mathcal{B}$
$\Phi^c \in \mathbb{R}^+$	Volume coefficient of container type $c \in \mathcal{C}$
$v^c \in \mathbb{R}^+$	Weight of container type $c \in \mathcal{C}$
$q_b \in \mathbb{R}^+$	Weight capacity of block $b \in \mathcal{B}$
$a^{tc} \in \mathbb{N}$	Number of available containers of type $c \in \mathcal{C}$ for transport $t \in \mathcal{T}$.

With this, the model can be formulated as seen below.

$$\texttt{Max Z} = \sum_{b \in \mathcal{B}} \sum_{t \in \mathcal{T}} \sum_{c \in \mathcal{C}} f^{tc} y_b^{tc} \tag{2.1}$$

`Subject to:`

$$\mathbf{w} \in \mathcal{W} \tag{2.2}$$

$$\sum_{t \in \mathcal{T}_p^{ON}} \sum_{c \in \mathcal{C}} v^c y_b^{tc} = w_{bp} \qquad \forall b \in \mathcal{B}, p \in \mathcal{P} \tag{2.3}$$

$$\sum_{t \in \mathcal{T}_p^{ON}} \sum_{c \in \mathcal{C}^{20}} y_b^{tc} \leq k_b^{20} \qquad \forall b \in \mathcal{B}, p \in \mathcal{P} \tag{2.4}$$

$$\sum_{t \in \mathcal{T}_p^{ON}} \sum_{c \in \mathcal{C}} \Gamma^c y_b^{tc} \leq k_b^{TEU} \qquad \forall b \in \mathcal{B}, p \in \mathcal{P} \tag{2.5}$$

$$\sum_{t \in \mathcal{T}_p^{ON}} \sum_{c \in \mathcal{R}^{20}} y_b^{tc} \leq r_b^{Slot} \qquad \forall b \in \mathcal{B}, p \in \mathcal{P} \tag{2.6}$$

$$\sum_{t \in \mathcal{T}_p^{ON}} \sum_{c \in \mathcal{R}} \tfrac{1}{2} \Gamma^c y_b^{tc} \leq r_b^{Cell} \qquad \forall b \in \mathcal{B}, p \in \mathcal{P} \tag{2.7}$$

$$\sum_{t \in \mathcal{T}_p^{ON}} \sum_{c \in \mathcal{C}} \Phi^c y_b^{tc} \leq h_b \qquad \forall b \in \mathcal{B}, p \in \mathcal{P} \tag{2.8}$$

$$\sum_{d \in \mathcal{P}} \sigma_{bp}^d = 1 \qquad \forall b \in \mathcal{B}, p \in \mathcal{P} \tag{2.9}$$

$$y_b^{tc} \leq a^{tc} \sigma_{bp}^{d^t} \qquad \forall b \in \mathcal{B}, c \in \mathcal{C}, p \in \mathcal{P}, t \in \mathcal{T}_p^{ON} \tag{2.10}$$

$$\sum_{b \in \mathcal{B}} y_b^{tc} \leq a^{tc} \qquad \forall t \in \mathcal{T}, c \in \mathcal{C} \tag{2.11}$$

$$0 \leq w_{bp} \leq q_b \qquad \forall b \in \mathcal{B}, p \in \mathcal{P} \tag{2.12}$$

$$y_b^{tc} \in \mathbb{N} \qquad \forall b \in \mathcal{B}, t \in \mathcal{T}, c \in \mathcal{C} \tag{2.13}$$

$$\sigma_{bp}^d \in \{0, 1\} \qquad \forall b \in \mathcal{B}, d \in \mathcal{P}, p \in \mathcal{P} \tag{2.14}$$

Objective (2.1) maximises the value of the containers loaded, where f^{tc} is a general function describing the value of each loaded container. This parameter can be changed to accommodate different objectives. In the case of revenue optimisation, it is

$$f^{tc} = u^{tc} \tag{2.15}$$

where u^{tc} is the revenue of container type $c \in \mathcal{C}$ of transport $t \in \mathcal{T}$ (see Section 2.6 for a description on how the revenue data is generated). In the case of intake optimisation, the function is

$$f^{tc} = \Gamma^c p^t \tag{2.16}$$

Here Γ^c is the TEU coefficient and p^t is the number of ports visited by the transport. This favours the selection of containers for long-haul transport. Constraint (2.2) ensures that the vessel is seaworthy, and fulfils all the limits outlined in the vessel profile.[1] Constraint (2.3) links the weight variable w_{bp} with the weight of the containers stowed. Equations (2.4)-(2.8) are the block capacity constraints. For each block of the vessel we distinguish between a TEU and a 20-foot capacity. This is due to the layout of the vessel since not all the blocks can stow 20-foot containers. In most of the cases k_b^{20} is, however, equal to k_b^{TEU}. Constraints (2.4) and (2.5) ensure that both of these capacities are satisfied. Similar to the TEU capacity constraints, the reefer constraints are split into a slot capacity constraint and a cell capacity constraint. Each reefer container needs to be plugged into an electrical output to get power for the cooling unit and r_b^{Slot} describes the number of such slots in a block b. A reefer cell is a cell where one of the slots has a reefer plug, and this number is denoted by r_b^{Cell}, if both slots in a reefer cell have a reefer plug, then $r_b^{Slot} = 2r_b^{Cell}$. In more general terms $r_b^{Cell} \leq r_b^{Slot} \leq 2r_b^{Cell}$ will be satisfied for all blocks. Constraint (2.6) restricts the number of reefer slots that can be used, and (2.7) is the reefer cell capacity constraint. Here we multiply with $\frac{1}{2}\Gamma^c$ as a 40-foot container occupies a full cell, whereas a 20-foot container occupies half a cell, and due to (2.6) the slot capacity will be satisfied. Constraint (2.8) limits the total volume of the containers in a block. This is needed in order to account for the fact that high-cube containers are higher and thus use more than a simple TEU capacity.

Constraints (2.9) and (2.10) are the block assignment constraints and together they enforce that no overstowage can occur. Specifically, constraint (2.9) ensures that exactly one port is assigned as the discharge port for every port. Constraint (2.10) makes sure that containers from

[1] For a complete description of the polyhedron $\mathcal{W}$ see Section 2.A.

a transport t can only be stowed in a block if the assigned destination matches the transports destination, d^t, during the full journey. Constraint (2.9) does not enforce that if $\sigma_{bp'}^{d'} = 1$ for a block b then all the ports until d' must have d' as discharge port. This will, however, be the case anyway since constraint (2.10) is posted for every port, which effectively disallows the stowage of containers with another ports of destination.

Constraint (2.11) ensures that no more containers than the ones expected in the cargo-flow can be selected for stowage. Finally (2.12)-(2.14) defines the variables domain, and (2.12) enforces the block weight capacity limit. The model includes ballast water within the set $\mathcal{W}$, but only as a way to fix stability issues, thus the ballast water is not minimised as part of the objective as in Delgado (2013).

The block stowage requirement is handled similar to how Ambrosino et al. (2015b) and Ambrosino et al. (2015a) handle it. The main difference is that we strictly enforce that overstowage cannot happen, where it is part of the objective in the MP-MBPP described in Ambrosino et al. (2015b) and Ambrosino et al. (2015a).

2.5 Solution Method

The model presented in Section 2.4 is in practice intractable to solve using standard MIP solvers (As shown by Table 2.1). Having integer and binary variables most commonly means that extra effort is needed for a MIP solver, and thus the σ_{bp}^{d} variables have a large impact on the effectiveness of the MIP model. Moreover, big-M constraints, as the ones in (2.10) deteriorate the LP-bound. The basic idea of the forthcoming matheuristic is to use a heuristic to fix the block assignment and effectively remove the σ_{bp}^{d} variables from the model. Additionally the equality requirement in (2.3) enforces a correspondence between the container weight variables and the total weight of a block. Such constraints often require extra effort from MIP solvers, thus the relaxation of these constraints could speed up the execution.

The proposed matheuristic is a three-phase procedure as illustrated in Figure 2.6. The algorithm takes as input a vessel profile, a vessel service, and an expected cargo flow and returns a feasible solution. It is a hierarchical algorithm, as the flow of information only goes downward, and the lower stages do not feed information back to the upper stages. Phase I heuristically determines the block assignments at each port. This then becomes a variable fixing of the σ_{bp}^{d} variables in the second phase. In Phase II the weight equality in (2.3) is relaxed, as is the integrality requirement of the cargo assignment variables (y_b^{tc}). The weight variables w_{bp} are used in the stability constraints. Therefore, to ensure the stability of the vessel w_{bp} must correspond to the actual weight stowed in a block. This exact correspondence is broken because of the relaxation of (2.3) in Phase II. The solution might then result in an unstable vessel. This is corrected in Phase III which receives as input from Phase II the values of the y_b^{tc} variables and solves a problem where the y_b^{tc} values are used as an upper bound on the cargo load, while also ensuring the solution is feasible wrt. to stability and integrality requirements.

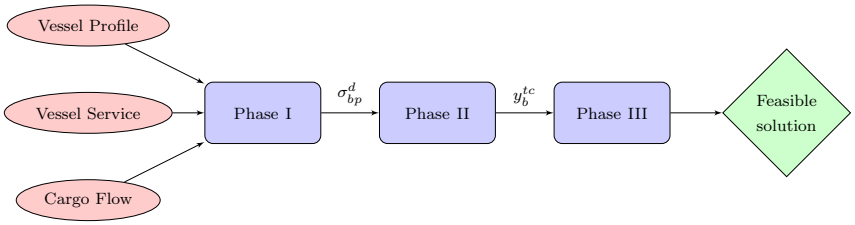


Figure 2.6: Flow chart outlining the flow of the matheuristic

2.5.1 Phase I

In the first phase, *schedules* are generated. A schedule s can be thoroughly described by the following parameter

$$b^d_{sp} = \begin{cases} 1, & \text{if schedule } s \text{ assigns destination } d \in \mathcal{P} \text{ as the discharge port when in port } p \in \mathcal{P} \\ 0, & \text{otherwise.} \end{cases}$$

A schedule s defines a block assignment for a full rotation, and is similar to the variables σ^d_{bp}, with the difference that σ^d_{bp} is for a given block. The first phase of the matheuristic must essentially do two things; generate schedules, and assign schedules to blocks.

When generating schedules, the goal is to find good schedules that fit the attributes of a specific type of block. We do so by considering the capacity resource and the possible assignments of container types. For every block, a schedule is generated by solving a longest path problem in a directed acyclic graph. Figure 2.7 is an example graph with 3 ports, the label on the nodes is the port symbolised by the node.

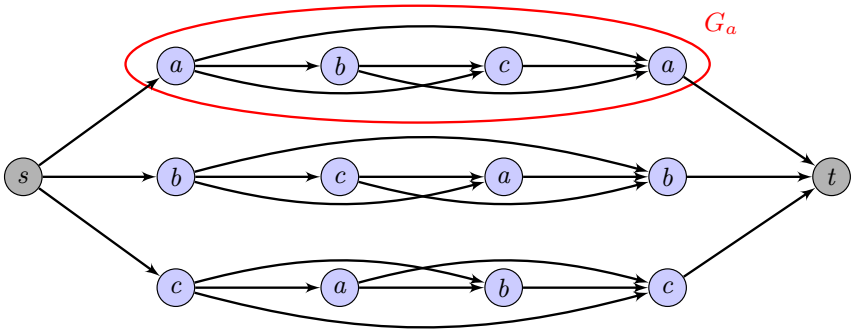


Figure 2.7: An example graph G of 3 ports, with the subgraph G_a highlighted.

An edge in the graph G describes a block assignment and a s-t path in G describes a schedule. If a path in G uses the edge (a, c), the corresponding schedule assigns discharge port c at port a. At all the ports in between a and c this assignment must be respected, and thus c is assigned as the discharge port for all these ports as well. Therefore, for the path (s, a, b, a, t) in G the corresponding schedule, s', assigns b as the discharge port when in port a, and discharge port a is assigned for port b and c. In terms of the parameter $b^b_{s'a}$, this corresponds to the following

$$b^b_{s'a} = 1, \qquad b^a_{s'b} = 1, \qquad b^a_{s'c} = 1$$

Let G be a weighted directed acyclic graph with $V(G)$ as the vertex set and $E(G)$ as the set of edges. The graph G consists of a source s, a sink t, and $|P|$ mutually disjoint subgraphs, G_p, such that

$$V(G) \setminus \{s,t\} = \bigcup_{p \in P} V(G_p) \quad \wedge \quad \bigcap_{p \in P} V(G_p) = \emptyset$$

Each node, i, in G_p symbolises a port in the rotation, this correspondence is described by the function $p(i)$ (see Figure 2.7 where the subgraph G_a is highlighted). The subgraph $G_{p'}$ contains exactly $|P|+1$ vertices. Let i_n be the n'th vertex in a topological sorting of the vertices in $G_{p'}$, then

$$p(i_n) = \begin{cases} p', & n = 1 \\ p(i_{n-1}) + 1, & 2 \leq n \leq |P|+1. \end{cases}$$

Where p' is a port in the rotation and $p'+1$ is the port visited immediately after p'.

Let $t_p(i)$ be the position of the vertex i in a topological sorting of the vertices of G_p, then let $\delta_p^+(i)$ describe the set of vertices, j, where $t_p(i) < t_p(j)$. The edge set of G_p can now formally be defined as

$$\begin{aligned} \delta_p^+(i) &= \{j \in V(G_p) \mid t_p(i) < t_p(j)\} && \forall i \in V(G_p), p \in P \\ E(G_p) &= \left\{(i,j) \mid i \in V(G_p), j \in \delta_p^+(i)\right\} && \forall p \in P \end{aligned}$$

Along with the edges from the subgraphs, G contains edges that connects the subgraphs to the source and the sink. Let $E(G_{base})$ be this set of edges defined as

$$E(G_{base}) = \{(s,j) \mid \exists p : t_p(j) = 1\} \cup \{(i,t) \mid \exists p : t_p(i) = |P|+1\}$$

With this the edge set of G can be properly defined as

$$E(G) \setminus E(G_{base}) = \bigcup_{p \in P} E(G_p)$$

The weight w_{ij}, for a given edge (i,j), represents the maximum number of containers that can be transported on the edge. The weights are heuristically set following the procedure described in Algorithm 1.

Algorithm 1 takes as input graph G, demand D, value matrix F, the block TEU (τ) and reefer TEU capacity (τ^r). The algorithm returns the edge weight matrix W. For each block b the TEU and reefer capacities can be defined as follows:

$$\tau = \max(k_b^{20}, k_b^{TEU}), \qquad \tau^r = \max(r_b^{Slot}, 2r_b^{Cell})$$

For the edges in $E(G_{base})$ the weight is 0, and therefore the loop in line 1 excludes these edges. Line 2-4 initialises the remaining TEU and reefer capacity for the edge (i,j). For an edge (i,j), the loop in lines 5 - 21 iterates over all the intermediate ports in the same order as they are visited, and calculate the weight of this edge. Here, $P_{p(i)}^{p(j)}$ is the set of ports visited between port $p(i)$ and port $p(j)$, including $p(i)$ but excluding $p(j)$. In line 7 the container types are sorted by value in descending order since we want to assign the most profitable containers first. For each container type c, we then have to calculate the available capacity. Lines 9-13 calculate the capacity of the specific type. If it is a reefer container we both need to account for the remaining

Algorithm 1 `Calculate` w_{ij}

Input: G, D, F, τ, τ^r,
1: **for all** $(i,j) \in E(G) \setminus E(G_{base})$ **do**
2: $w_{ij} \leftarrow 0$
3: $\Delta\tau \leftarrow \tau$
4: $\Delta\tau^r \leftarrow \tau^r$
5: **for all** $k \in P_{p(i)}^{p(j)}$ **do**
6: $t \leftarrow$ The transport $\langle k, p(j)\rangle$
7: $sort\mathcal{C} \leftarrow$ Container types of transport t sorted by descending order of f^{tc}
8: **for all** $c \in sort\mathcal{C}$ **do**
9: **if** $c \in \mathcal{R}$ **then**
10: capacity $\leftarrow \min(\Delta\tau,\ \Delta\tau^r)$
11: **else**
12: capacity $\leftarrow \Delta\tau$
13: **end if**
14: load $\leftarrow \min\left(d^c_{kp(j)}, \lfloor \frac{\text{capacity}}{\Gamma^c} \rfloor\right)$
15: $w_{ij} \leftarrow w_{ij} + f^{tc}\text{load}$
16: $\Delta\tau \leftarrow \Delta\tau - \Gamma^c\text{load}$
17: **if** $c \in \mathcal{R}$ **then**
18: $\Delta\tau^r \leftarrow \Delta\tau^r - \Gamma^c\text{load}$
19: **end if**
20: **end for**
21: **end for**
22: **end for**
23: **return** W

reefer capacity, as well as the remaining TEU capacity, otherwise the remaining TEU capacity is sufficient. Line 14 determines the number of containers that can be assigned the edge. Here $d^c_{kp(j)}$ is the demand from port k to port $p(j)$ of container type c. The number of containers that can be assigned the edge is the minimum between the demand and the calculated capacity. We divide the capacity by Γ^c, the TEU coefficient, to have the actual number of containers. Line 15 updates the edge weight, while lines 16-19 update the remaining TEU capacity, $\Delta\tau$, and the remaining reefer capacity, $\Delta\tau^r$.

The edge weights are set in such a way that the longest path in the graph corresponds to a block assignment where the structure of the demand is taken into account. For instance, ports with a comparatively high number of unloading containers will be chosen as the destination for more origin ports. The longest path problem is solved for each block. The assigned cargo is then subtracted from the demand matrix to account for the cargo the generated schedule can carry. The edge weights calculations are based on the demand and thus the edge weights are updated for every block. Therefore, the longest path differs from block to block.

The graph G is a directed acyclic graph, and the longest path problem can be solved efficiently by topological sorting the vertexes and using a slightly modified version of Dijkstra's algorithm, such that the longest path is found instead of the shortest.

2.5.2 Phase II

In the second phase the weight equality in (2.3) is relaxed, and a mixed integer programming model assigns containers to blocks such that the block assignment is satisfied.

Let the parameter $\widehat{\sigma}_{bp}^{d}$ describes the block assignment from Phase I as follows:

$$\widehat{\sigma}_{bp}^{d} = \begin{cases} 1, & \text{If block } b \in \mathcal{B} \text{ is assigned destination } d \in \mathcal{P} \text{ at port } p \in \mathcal{P} \\ 0, & \text{Otherwise.} \end{cases}$$

Hence the interpretation of $\widehat{\sigma}_{bp}^{d}$ is similar to that of the variable σ_{bp}^{d}. The integrality constraint of y_b^{tc} are relaxed resulting in the following Phase II model:

$$\texttt{Max} \quad (2.1)$$

`Subject to:`

$$(2.2)$$
$$(2.4) - (2.8)$$
$$(2.11) - (2.12)$$

$$\sum_{t \in \mathcal{T}_p^{ON}} \sum_{c \in \mathcal{C}} v^c y_b^{tc} \leq w_{bp} \qquad \forall b \in \mathcal{B}, p \in \mathcal{P} \qquad (2.17)$$

$$y_b^{tc} \leq a^{tc} \widehat{\sigma}_{bp}^{d_t} \qquad \forall b \in \mathcal{B}, c \in \mathcal{C}, p \in \mathcal{P}, t \in \mathcal{T}_p^{On} \qquad (2.18)$$

$$y_b^{tc} \geq 0 \qquad \forall b \in \mathcal{B}, t \in \mathcal{T}, c \in \mathcal{C} \qquad (2.19)$$

Constraint (2.17) is a relaxation of the weight setting constraint (2.3), while (2.18) ensures that the block assignment from Phase I is satisfied. We can omit constraint (2.9) since it is implicitly satisfied by the fixed variable assignment $\widehat{\sigma}_{bp}^{d}$. The y_b^{tc} variables are enforced to be non-negative in (2.19). Additionally, the integrality requirement is relaxed to speed up the execution of Phase II.[2]

Due to the relaxation of (2.3) and (2.19), this model might overestimate the number of loaded containers, as the weight variables used in the stability calculation is not strictly enforced to match the actual weight in the block.

Experimental evaluation has shown that this model can find high-quality solutions fast. Closing the final gap between the lower and upper bound and proving optimality is, however, time-consuming. For this reason, we impose an optimality tolerance η, meaning that the solution process is stopped once the gap between the lower and upper bound reaches this tolerance value.

2.5.3 Phase III

Phase III uses the solution from Phase II as a starting point and ensures that the final vessel configuration is stable, while enforcing the integrality requirement of the cargo assignment variables.

First let $\widehat{y}_b^{tc}$ describe the container placement solution from Phase II, and define the variable u_b^{tc} $\in \mathbb{N}$ as the number of containers of type c in transport t to be stowed in block b. To decrease

[2] Due to a binary indicator variable in the stability constraints, the problem is still a MIP.

the computational effort needed to ensure the feasibility of the solution, the u_b^{tc} variables are restricted to be less than or equal to $\widehat{y}_b^{tc}$.

The model solved in Phase III is the following

$$\texttt{Min Z} = \sum_{b \in \mathcal{B}} \sum_{t \in \mathcal{T}} \sum_{c \in \mathcal{C}} f^{tc} \widehat{y}_b^{tc} - f^{tc} u_b^{tc} \tag{2.20}$$

Subject to:

$$\text{(2.2)}$$
$$\text{(2.12)}$$
$$\sum_{t \in \mathcal{T}_p^{On}} \sum_{c \in \mathcal{C}} v^c u_b^{tc} = w_{bp} \qquad \forall b \in \mathcal{B}, p \in \mathcal{P} \tag{2.21}$$
$$0 \leq u_b^{tc} \leq \widehat{y}_b^{tc} \qquad \forall b \in \mathcal{B}, t \in \mathcal{T}, c \in \mathcal{C} \tag{2.22}$$
$$u_b^{tc} \in \mathbb{N} \qquad \forall b \in \mathcal{B}, t \in \mathcal{T}, c \in \mathcal{C} \tag{2.23}$$

The objective function (2.20) minimises the difference between the solution from Phase II and the final solution. This corresponds to removing the minimum number of containers, in terms of objective value. Constraint (2.21) sets the w_{bp} variable, forcing it to be equal to the weight stowed in block b at port p. Constraint (2.22) restricts the u_b^{tc} variables to be less than or equal to $\widehat{y}_b^{tc}$. The solution from Phase II respects the capacity constraints, and by only removing containers these constraints will still be satisfied. Therefore the capacity constraints can be disregarded in this phase. Finally (2.23) defines the domain for the cargo assignment variables.

2.6 Data Description

The vessel used for the experiments is operated by our industrial collaborator and has approximately 50 blocks and a TEU capacity of ca. 15,000 TEUs. The vessel data is confidential, thus only the vessel capacity can be made public.

Based on this vessel, we have generated a benchmark of 170 demand instances. All instances have been grouped by the number of ports to visit, which ranges between 4 and 20. Each group consists of 10 instances with varying demands. The ports are selected randomly from a list of 28 ports that are visited on the Europe-Asia services operated by our industrial collaborator. The order the ports are visited in is preserved, and the port-to-port demand and port drafts are based on data from LinerLib (Brouer et al., 2014). The demand from the LinerLib does not include transhipents (as those are part of the network design problem), however, it can be used as indication. To make sure that the demand corresponds to a cargo-flow which tends to utilize the full vessel capacity, we multiply the demand of each service leg by a scalar $\gamma = \textit{number of ports} \cdot \textit{nominal intake}/\textit{LinerLib total demand}$. Successively the demand of each leg is decomposed into a demand for each container type by predefined probabilities. The probability for a container being a forty foot container is 40%, the reefer probability is 10%, and high-cube 10%. These probabilities are based on numbers from our industrial collaborator. The

probabilities for a container having weight class w is denoted $P(w)$ and a reasonable distribution could be the following.

$$P(3) = 5\%, \quad P(6) = 10\%, \quad P(9) = 15\%, \quad P(14) = 25\%, \quad P(21) = 25\%, \quad P(27) = 20\%$$

These probabilities are sufficient to give a full probability distribution for all container types. The probability for generating a 40 foot, high-cube non-reefer container of weight 21 is, for example,

$$0.40 \cdot 0.10 \cdot (1 - 0.10) \cdot 0.25 = 0.9\%$$

Finally, using the value of a dry 40-foot container from the LinerLib, we calculate the revenue of each container type as described in Delgado (2013).

2.7 Computational Results

Four different solution methods have been tested and compared.

- **CMPBSMIP - CMPBS Mixed Integer Programming model**
  The mixed integer programming model as presented in Section 2.4.
- **CMPBSHLP - CMPBS Heuristic Linear**
  The 3-phase matheuristic proposed in Section 2.5.
- **CMPBSHIP - CMPBS Heuristic Integer**
  This is a version of the 3-phase matheuristic where we do not relax the integrality constraints on the y_b^{tc} variables of Phase II.
- **2-CMPBSH - 2 Phased CMPBS Heuristic**
  This is a version of the matheuristic where both the integrality constraints of the y_b^{tc} variables and the equality of constraint (2.3) are not relaxed. In this specific case, Phase III is no longer needed.

All methods have been implemented in JAVA 1.7 and are tested using a 2.30 GHz Intel Xeon E5 processor. CPLEX v. 12.6.1 is used to solve the MIP models in Section 2.4, Section 2.5.2 and Section 2.5.3 and each model is solved using 8 threads. For CMPBSMIP, a time limit of 1 hour has been used. For the three heuristics, the time limit is 10 minutes. For all methods, an optimality tolerance $\eta = 0.05\%$ was used.

Since only very few optimal solutions of the CMPBS can be calculated, we propose an upper bound to estimate the quality of the solution methods. The upper bound was computed by removing the block stowage requirement from the cargo mix model (constraints (2.9)-(2.10)). The best dual bound after 30 minutes of solving time is used as the upper bound for the CMPBS, and it is referred to as UB in the following.

2.7.1 Intake optimisation

For the intake optimisation, Table 2.1 shows aggregated results for the 4 tested method. Here $|P|$ describes the number of ports. #Sol is the number of instances where a feasible solution is found, the $\bar{x}$ columns are the average solution value, lastly $\bar{t}$ is the average execution time in seconds. The solution values reported for CMPBSMIP are from the best feasible solutions found within the time limit. Thus the values are not ensured to be better than that of the heuristics.

Table 2.1 shows that the CMPBSMIP model is too complex to solve real world sized instances. For instances with more than 6 ports the quality of the solutions are too bad, and for instances, with 11 or more ports the MIP model fails to even find a feasible solution within an hour. In total, only 2 instances are solved to optimality, both instances only having 4 ports. CMPBSHLP finds good quality solutions in just 14 seconds. Comparing with CMPBSHIP it can be seen that the relaxation of the cargo variables y_b^{tc} does not have a significant impact on solution quality, but the computational times are increased considerably. The best solutions are found with 2-CMPBSH. The method, however, is 8 times slower than CMPBSHLP. All three heuristic methods find feasible solutions for all the tested instances.

Table 2.1: Intake optimisation: Aggregated results

	UB	**CMPBSMIP**			**CMPBSHLP**		**CMPBSHIP**		**2-CMPBSH**	
$\|P\|$	$\bar{x}$	#Sol	$\bar{x}$	$\bar{t}$	$\bar{x}$	$\bar{t}$	$\bar{x}$	$\bar{t}$	$\bar{x}$	$\bar{t}$
4	53265	10	46352	2957.5	50351	1.3	50551	5.6	50851	4.5
5	68153	8	58514	3600.0	62917	2.1	63090	5.5	63625	6.6
6	85935	5	41543	3600.0	81209	2.0	81472	9.4	81834	17.7
7	102409	4	39484	3600.0	93191	3.9	93490	20.2	94387	27.5
8	122125	7	48448	3600.0	112787	5.2	113198	19.9	113339	23.4
9	131360	1	53942	3600.0	122022	4.4	122375	28.9	122742	38.2
10	149285	3	54073	3600.0	138485	7.5	138996	28.1	139113	55.6
11	170912	0	-	3600.0	155353	7.4	155884	48.0	155967	61.8
12	185836	0	-	3600.0	168760	17.4	169457	63.4	169473	88.9
13	201501	0	-	3600.0	185299	17.4	186007	129.6	186024	100.6
14	212244	0	-	3600.0	196018	13.2	196596	66.5	197172	97.3
15	231770	0	-	3600.0	212763	18.9	213544	82.8	213546	154.3
16	249557	0	-	3600.0	229117	23.2	229921	120.5	229933	187.0
17	265357	0	-	3600.0	245174	19.4	245936	135.0	246106	239.1
18	280448	0	-	3600.0	258001	43.2	258989	201.3	259060	238.2
19	296581	0	-	3600.0	273704	25.7	274680	171.7	274689	246.6
20	308392	0	-	3600.0	285586	24.6	286621	125.5	286935	281.7
Average	183243		48908	3562.2	168867	13.9	169459	74.2	169694	109.9

For the 10 instances with 4 ports, Table 2.2 compares the upper bound with the solution and the bound after 5 hours of solving the model together with the CMPBSHLP. The '*' symbol denotes instances where the model execution was terminated due to the optimality tolerance $\eta = 0.05\%$. A time limit of 1 hour or 5 hours does not change the number of optimal solutions found, but for the non-optimal instances, the solution quality is improved. For all the instances with 4 ports, when given the CMPBSMIP 5 hours instead of 1 hour, the model finds better solutions and outperforms the CMPBSHLP heuristic with respect to solution quality. For the two optimal solutions, the table shows that the upper bound does not differ much from the optimal solution. For the rest of the instances, the upper bound is comparable to the bound achieved after 5 hours of solving the compact model. This indicates that the objective value found by the upper bound

method is close to the optimal solution.

Table 2.2: Upper bound analysis

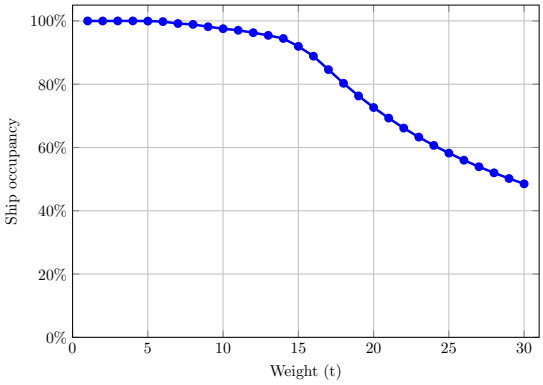
		UB	CMPBSMIP			CMPBSHLP	
$\|P\|$	i	x	x	x_{bound}	t	x	t
4	0	41894	41742	41893	18000	39014	1.1
4	1	54016	**53992***	54017	690	51898	0.8
4	2	59349	57963	59361	18000	55012	2.6
4	3	59314	59202	59314	18000	57387	1.4
4	4	57242	57081	57225	18000	53713	1.0
4	5	45185	**45165***	45186	311	44339	1.2
4	6	56634	56046	56632	18000	54556	1.7
4	7	47172	47115	47172	18000	45833	1.1
4	8	60146	59260	59832	18000	57001	1.1
4	9	51693	51062	51693	18000	44756	1.4
Average		53265	52863	53233	14500	50351	1.3

Table 2.3 evaluates the quality of the schedules obtained from solving the longest path problem in Phase I. The graph based method is compared with a random method, where schedules are generated at random. The table reports the final solution after Phase III. The solution in the Graph column is identical to the CMPBSHLP results from Table 2.1. To account for the randomness of the Random method, each instance is solved 10 times with a different set of random schedules. $\bar{x}^b$ reports the average of the best solutions for the considered instance group, and $\bar{x}$ is the average of all solutions for the instance group. The table shows that the graph based method substantially outperforms the random method. This is also to be expected as the graph based method cleverly generates schedules based on the demand.

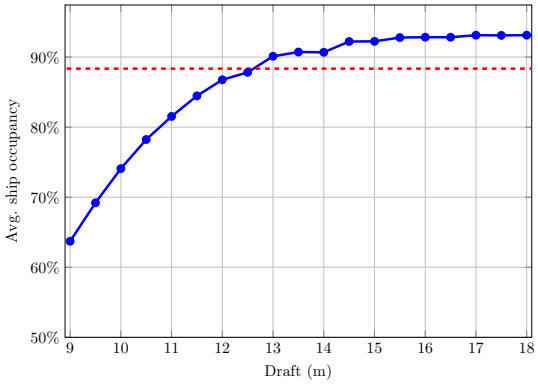
Table 2.3: Analysis of the quality of the schedules from Phase I.

	UB	Graph	Random	
$\|P\|$	$\bar{x}$	$\bar{x}$	$\bar{x}$	$\bar{x}^b$
4	53265	50351	33383	37037
5	68153	62917	37632	42985
6	85935	81209	49896	55470
7	102409	93191	56833	61875
8	122125	112787	70806	78137
9	131360	122022	72862	81859
10	149285	138485	81115	90149
11	170912	155353	96576	105861
12	185836	168760	109886	120247
13	201501	185299	112737	124068
14	212244	196018	114017	126599
15	231770	212763	124640	139470
16	249557	229117	137357	151468
17	265357	245174	141310	158254
18	280448	258001	155261	169868
19	296581	273704	166183	182065
20	308392	285586	162003	179398
Average	183243	168867	101323	112048

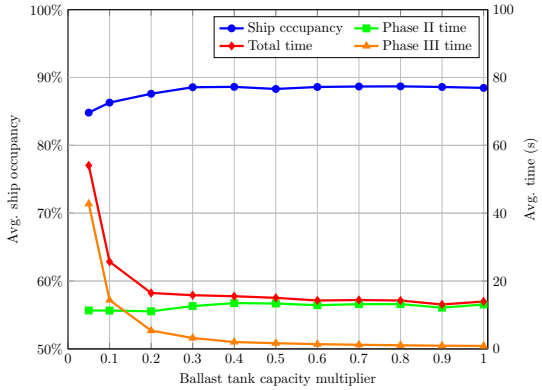
Figure 2.8 shows an analysis of how different factors impact the ship occupancy. Solutions to the CMPBS tend to load a high number of light containers (as further described in Section 2.7.2). Figure 2.8a illustrates the occupancy rate of the vessel when the CMPBS is allowed to only load



(a) Maximum achievable ship occupancy when only containers of a certain weight class are available.



(b) Average ship occupancy, over all instances, when all ports have the same draft.



(c) Average ship occupancy, over all instances, when limiting the ballast water allowed.

Figure 2.8: Ship occupancy analysis

containers of a certain weight class. It can be easily seen that when we force the model to use the heavier containers, the ship occupancy drops resulting in a smaller intake.

In a similar fashion, the graph in Figure 2.8b illustrates the impact of draft restrictions at port. The graph shows the average ship occupancy over all instances, where the draft has been fixed to a single value for all ports. A small draft restricts the total displacement of the vessel, which means fewer containers can be loaded. When the draft is 15.5m or above the impact of the draft constraint is minor, and the ship occupancy is nearly constant.

It is worth noticing that the use of ballast tanks, not only helps satisfy the vessel seaworthiness requirements, they also impact the solution time of the method (Pacino et al., 2012). The variables that refer to the ballast tanks (see Appendix 2.A) are continuous and behave as slack variables on the stability constraints. We show this in Figure 2.8c which depicts both the impact with respect to the ship occupancy and the solution time for the CMPBSHLP method. Each of the ballast tanks has a maximum capacity, this capacity is multiplied by a number between 0 and 1, to analyse its impact. The graph shows that the ship occupancy slightly increases the bigger the capacity. This is expected as the more water that can be loaded into the ballast tanks

the easier it is to satisfy stability issues. Also with respect to solution time the results are as expected. The graph shows that the higher the tank capacity, the less time is needed to solve the problem. The impact on the runtime of Phase II is minor. For low tank capacities, more time is spent in Phase III where the stability constraints are strictly enforced.

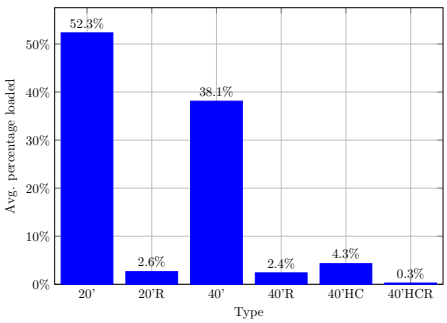
2.7.2 Revenue Optimisation

Table 2.4 shows aggregated results for the 4 tested methods when optimising revenue, instead of intake. The $\bar{x}$ columns show the average revenue in millions of USD, the rest of the columns are similar to those of Table 2.1 described in Section 2.7.1. As can be seen, the solution methods behave, in terms of solution quality and runtime performance, in the same way as for intake optimisation. The same conclusions can thus be drawn.

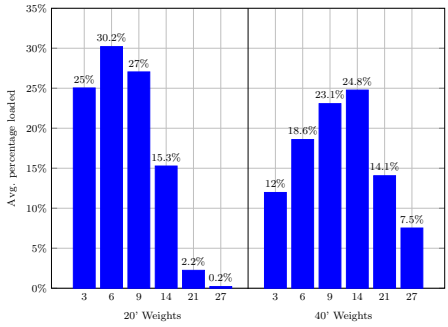
Table 2.4: Revenue optimisation: Aggregated results

	UB	**CMPBSMIP**			**CMPBSHLP**		**CMPBSHIP**		**2-CMPBSH**	
$\|P\|$	$\bar{x}$ (10^6 \$)	#Sol	$\bar{x}$ (10^6 \$)	$\bar{t}$	$\bar{x}$ (10^6 \$)	$\bar{t}$	$\bar{x}$ (10^6 \$)	$\bar{t}$	$\bar{x}$ (10^6 \$)	$\bar{t}$
4	29.93	10	29.77	2512.2	28.70	0.9	28.81	2.8	28.95	6.3
5	37.13	10	35.27	3007.2	35.88	1.3	36.00	4.9	36.03	7.5
6	36.99	9	26.76	3600.0	35.15	2.3	35.30	10.1	35.33	18.8
7	40.89	7	16.90	3600.0	39.07	2.9	39.22	20.5	39.37	39.9
8	41.79	7	13.65	3600.0	39.52	4.6	39.69	26.0	39.75	39.7
9	42.68	2	6.08	3600.0	40.54	3.6	40.71	27.6	40.92	65.0
10	43.10	1	1.96	3600.0	40.67	4.3	40.83	20.9	40.96	87.8
11	47.03	1	2.82	3600.0	44.19	10.0	44.39	47.6	44.35	134.9
12	48.19	0	-	3600.0	45.27	9.5	45.46	62.9	45.50	123.9
13	49.67	0	-	3600.0	46.65	13.6	46.84	59.1	47.08	183.4
14	50.50	0	-	3600.0	47.56	8.5	47.73	98.3	47.21	387.0
15	49.40	0	-	3600.0	46.57	17.1	46.73	91.6	46.94	170.7
16	50.26	0	-	3600.0	47.53	20.2	47.73	132.6	47.72	407.0
17	51.93	0	-	3600.0	48.16	18.4	48.34	103.9	48.61	218.5
18	53.98	0	-	3600.0	49.81	25.7	50.00	94.4	50.10	268.5
19	55.51	0	-	3600.0	51.47	33.5	51.68	163.2	49.80	370.7
20	51.83	0	-	3600.0	48.05	27.6	48.22	193.4	48.55	318.2
Average	45.93		16.65	3501.1	43.22	12.0	43.39	68.2	43.36	167.5

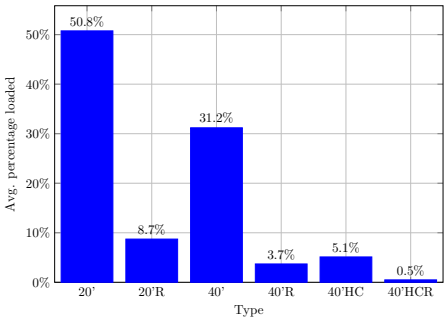
The difference between intake and revenue optimisation is clearer if we look at the type of cargo that is being loaded. Figure 2.9 shows a histogram of the containers occupying the vessel during the full rotation. The data is based on the average over all 170 instances. Figure 2.9a and Figure 2.9b is for the case of intake optimisation, and Figure 2.9c and Figure 2.9d is for the case of revenue optimisation. Figure 2.9a and Figure 2.9c show the container types loaded, here the label corresponds to the type of container, first the length, then reefer types are denoted with a R, and high-cubes with HC. For intake and revenue optimisation the distribution of container types is similar. The loading of reefer containers is expected to be low, even in revenue optimisation, as it is constrained by the available reefer plugs in the vessel. High-cubes tend to reduce capacity due to the extra volume, so it also makes sense that not many of them are loaded in either intake or revenue optimisation. Figure 2.9b and Figure 2.9d show the weight distribution of the $20'$ and $40'$ containers types loaded (excluding reefer and high-cubes). When optimising the intake, the value of a container does not dependent on the weight, and therefore the model has a tendency to load light and empty containers (3 tonnes). When loading light containers the weight capacity is of less importance, and thus more containers can be loaded in total. However, empty containers



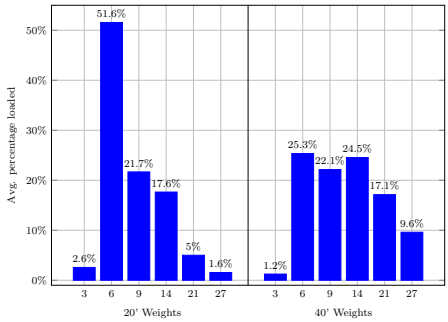
(a) Intake optimisation: Container load distribution



(b) Intake optimisation: Container weight load distribution



(c) Revenue optimisation: Container load distribution



(d) Revenue optimisation: Container weight load distribution

Figure 2.9: Containers loaded when optimising the intake and the revenue.

do not generate as much revenue, and therefore only a few empties are loaded when optimising revenue. Though heavier containers generate more revenue, they also reduce vessel capacity since the weight limits of the stacks are reached faster, as was shown in Figure 2.8a.

Finally in Table 2.5 we show the impact of the relaxation of the weight constraints in Phase II for the CMPBSHLP matheuristic. Here $\bar{x}$ is the solution value after the corresponding phase, and $\bar{t}$ is the average runtime of each phase. The column %Rem is the percentage of removed containers by Phase III in terms of objective value. In both cases, the most time is spent in Phase II, of which approximately 10% is spent in Phase III. As can be seen Phase III only removes in average ca. 0.75% of the containers in terms of objective values.

2.8 Conclusion

In this paper, the Cargo Mix Problem with Block Stowage has been introduced. The problem has been described thoroughly and formulated as a mathematical model. The mathematical model is not scalable, and cannot solve any of the real-life sized instances. To overcome this, a matheuristic has been developed. The method combines a 3-phase decomposition with variable fixing and heuristic assignments.

Table 2.5: Detailed results for CMPBSHLP when optimising intake and revenue

	Intake					Revenue				
	Phase II		Phase III			Phase II		Phase III		
$\|P\|$	$\bar{x}$	$\bar{t}$	$\bar{x}$	$\bar{t}$	%Rem	$\bar{x}$ (10^6 \$)	$\bar{t}$	$\bar{x}$ (10^6 \$)	$\bar{t}$	%Rem
4	50985	1.1	50351	0.3	1.35%	29.00	0.7	28.70	0.2	1.03%
5	63883	1.8	62917	0.3	1.53%	36.04	1.0	35.88	0.2	0.44%
6	82100	1.6	81209	0.4	1.18%	35.36	1.9	35.15	0.4	0.56%
7	94995	3.4	93191	0.5	2.03%	39.49	2.3	39.07	0.6	0.97%
8	113379	4.5	112787	0.7	0.53%	39.77	4.0	39.52	0.6	0.61%
9	122824	3.7	122022	0.7	0.66%	40.96	2.7	40.54	0.9	1.00%
10	139182	6.7	138485	0.8	0.50%	41.01	3.3	40.67	1.0	0.77%
11	156014	6.7	155353	0.7	0.42%	44.43	9.3	44.19	0.7	0.50%
12	169533	16.6	168760	0.9	0.46%	45.52	8.5	45.27	1.0	0.50%
13	186085	16.4	185299	1.0	0.42%	47.26	12.1	46.65	1.4	1.21%
14	197610	12.1	196018	1.1	0.85%	47.94	7.0	47.56	1.5	0.76%
15	213619	18.0	212763	0.9	0.40%	47.02	15.4	46.57	1.7	0.92%
16	230005	22.0	229117	1.2	0.39%	47.84	18.3	47.53	1.9	0.60%
17	246212	17.9	245174	1.5	0.42%	48.66	16.1	48.16	2.3	0.96%
18	259156	41.8	258001	1.4	0.45%	50.24	23.0	49.81	2.7	0.80%
19	274787	24.3	273704	1.4	0.40%	51.78	31.6	51.47	1.9	0.57%
20	287183	23.0	285586	1.5	0.57%	48.63	24.5	48.05	3.1	1.09%
Average	169856	13.0	168867	0.9	0.74%	43.59	10.7	43.22	1.3	0.78%

Three variations of the matheuristic were tested on 170 data instances, both when optimising revenue and intake. The results showed that the approach finds high-quality solutions in seconds and can scale to industrial size instances.

Though the described problem can be seen as a strategic planning tool, its analytical strength is better suited at the operational level e.g. in uptake management. Since the solution approach is based on mathematical models, it is easy to add extra constraints to e.g. perform what-if scenario analysis. To give an example, given the current state of a vessel, meaning its current load and remaining capacity, it is possible to use the CMPBS to identify an optimised number of containers to choose from a list of bookings. The impact of a booking could also be compared to an optimised cargo-mix to estimate the impact it will have on revenue. The authors believe that the mathematical models and the solution approach could easily be adapted to perform this kind of analysis within, for example, a decision support system. More so since the presented method can solve the CMPBS within seconds.

Possible areas of future research can focus on both optimal and heuristic methods. The decomposition currently adopted could be suitable for the implementation of a column generation approach where the schedules are dynamically generated. Further research, however, needs to be done for such an approach to be successful due to the complexity of the resulting reduced model.

Since the quality of the final solution heavily depends on the block assignment in Phase I, an idea for future research is to improve the way it is computed. This can be done by improving the way schedules are generated, and thus keeping the hierarchical nature of the algorithm, or by using an iterative procedure. In an iterative approach, the solution from one of the later phases is fed back to a previous phase to re-optimise, until some termination criterion is met. Within a decision support system setting, this approach is, however, not suitable for the current hierarchical heuristic since it would use an average of ca. 14 seconds for each iteration.

Furthermore, we plan to extend the problem by considering stochastic cargo flows, and we also plan to study how this work can be combined with other important problems in the liner shipping industry, e.g. service network design and cargo flow optimisation.

Acknowledgements

This work has been funded by the Danish Innovationsfonden under the GREENSHIP Project (1313-00005B-GREENSHIP).

References

Ambrosino, D. and A. Sciomachen (1998). "A Constraint Satisfaction Approach for Master Bay Plans". In: *Water studies series* 36.

Ambrosino, D., D. Anghinolfi, M. Paolucci, and A. Sciomachen (2009). "A new three-step heuristic for the master bay plan problem". In: *Maritime Econ Logistics* 11.1, pp. 98–120. ISSN: 1479-2931.

Ambrosino, D., D. Anghinolfi, M. Paolucci, and A. Sciomachen (2010). "An Experimental Comparison of Different Heuristics for the Master Bay Plan Problem". In: *Experimental Algorithms*. Ed. by P. Festa. Vol. 6049. Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 314–325.

Ambrosino, D., M. Paolucci, and A. Sciomachen (2015a). "A MIP Heuristic for Multi Port Stowage Planning". In: *Transportation Research Procedia* 10. 18th Euro Working Group on Transportation, EWGT 2015, 14-16 July 2015, Delft, The Netherlands, pp. 725–734. ISSN: 2352-1465.

Ambrosino, D., M. Paolucci, and A. Sciomachen (2015b). "Computational evaluation of a MIP model for multi-port stowage planning problems". In: *Soft Computing*, pp. 1–11. ISSN: 1433-7479.

Ambrosino, D., A. Sciomachen, and E. Tanfani (2004). "Stowing a containership: the master bay plan problem". In: *Transportation Research Part A: Policy and Practice* 38.2, pp. 81–99. ISSN: 09658564.

Avriel, M., M. Penn, N. Shpirer, and S. Witteboon (1998). "Stowage planning for container ships to reduce the number of shifts". In: *Annals of Operations Research* 76.1-4, pp. 55–71.

Botter, R. C. and M. A. Brinati (1992). "Stowage Container Planning: A Model for Getting an Optimal Solution". In: *Proceedings of the IFIP TC5/WG5.6 Seventh International Conference on Computer Applications in the Automation of Shipyard Operation and Ship Design, VII*. Amsterdam, The Netherlands, The Netherlands: North-Holland Publishing Co., pp. 217–229. ISBN: 0-444-89728-3.

Brouer, B. D., J. F. Alvarez, C. E. M. Plum, D. Pisinger, and M. M. Sigurd (2014). "A Base Integer Programming Model and Benchmark Suite for Liner-Shipping Network Design". In: *Transportation Science* 48, pp. 281–312. ISSN: 0041-1655.

Christensen, J. and D. Pacino (2017). "A matheuristic for the Cargo Mix Problem with Block Stowage". In: *Transportation Research Part E: Logistics and Transportation Review* 97, pp. 151–171. ISSN: 1366-5545.

Christiansen, M., K. Fagerholt, B. Nygreen, and D. Ronen (2007). "Chapter 4 Maritime Transportation". In: *Handbooks in Operations Research and Management Science*.

Delgado, A. (2013). "Models and Algorithms for Container Vessel Stowage Optimization".

Delgado, A., R. Jensen, and K. Janstrup (2012). "A constraint programming model for fast optimal stowage of container vessel bays". In: *European Journal of Operational Research* 220.1, pp. 251–261.

Ding, D. and M. C. Chou (2015). "Stowage Planning for Container Ships: A Heuristic Algorithm to Reduce the Number of Shifts". In: *European Journal of Operational Research*. ISSN: 03772217.

Dubrovsky, O., G. Levitin, and M. Penn (2002). "A genetic algorithm with a compact solution encoding for the container ship stowage problem". In: *Journal of Heuristics*, pp. 585–599.

Feng, C.-M. and C.-H. Chang (2008). "Optimal Slot Allocation in Intra-Asia Service for Liner Shipping Companies". In: *Maritime Economics & Logistics* 10.3, pp. 295–309. ISSN: 1479-2931.

Kang, J.-G. and Y.-D. Kim (2002). "Stowage planning in maritime container transportation". In: *Journal of the Operational Research Society* 53.4, pp. 415–426.

Li, F., C. Tian, R. Cao, and W. Ding (2008). "An integer linear programming for container stowage problem". In: *Computational Science–ICCS 2008*, pp. 853–862.

Pacino, D., A. Delgado, R. M. Jensen, and T. Bebbington (2011). "Fast Generation of Near-Optimal Plans for Eco-Efficient Stowage of Large Container Vessels". In: *Computational Logistics*. Ed. by J. W. Böse, H. Hu, C. Jahn, X. Shi, R. Stahlbock, and S. Voß. Vol. 6971. Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 286–301.

Pacino, D., A. Delgado, R. Jensen, and T. Bebbington (2012). "An Accurate Model for Seaworthy Container Vessel Stowage Planning with Ballast Tanks". English. In: *Computational Logistics*. Ed. by H. Hu, X. Shi, R. Stahlbock, and S. Voß. Vol. 7555. Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 17–32. ISBN: 978-3-642-33586-0.

Pacino, D. and R. Jensen (2012). "Constraint-based local search for container stowage slot planning". In: *Lecture Notes in Engineering and Computer Science* 2, pp. 1467–1472. ISSN: 2078-0958.

Reinhardt, L. B. and D. Pisinger (2012). "A branch and cut algorithm for the container shipping network design problem". In: *Flexible Services and Manufacturing Journal* 24, pp. 349–374. ISSN: 19366582.

Sciomachen, A. and E. Tanfani (2003). "The master bay plan problem: a solution method based on its connection to the three-dimensional bin packing problem". In: *IMA Journal of Management Mathematics* 14, pp. 251–269.

Wilson, I. and P. Roach (2000). "Container Stowage Planning: A Methodology for Generating Computerised Solutions". In: *The Journal of the Operational Research Society* 51.11, pp. 1248–1255.

Zurheide, S. and K. Fischer (2011). "A Revenue Management Slot Allocation Model with Prioritization for the Liner Shipping Industry". In: *Operations Research Proceedings 2010: Selected Papers of the Annual International Conference of the German Operations Research Society*. Ed. by B. Hu, K. Morasch, S. Pickl, and M. Siegle. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 143–148. ISBN: 978-3-642-20009-0.

2.A Stability Constraints

The stability constraints ensure that the vessel does not capsize or break, even in tough weather conditions. The constraints here uses similar notation as the one introduced in Delgado (2013),

and the interested reader is invited to look at the source for a more elaborate description.

The main difference between the set of constraints presented below and (6.25) - (6.43) of Delgado (2013), is the draft. Delgado (2013) does not consider draft limit for each port, thus making it possible for the vessel to have a greater displacement. The introduction of the LinerLib (Brouer et al., 2014) makes this data easily available, and the draft limit is therefore included in this work. Also, Delgado considers locations, where this work considers blocks.

The functions describing the metacentre, trim, draft and buoyancy force all depends non-linearly on the displacement. However, Delgado (2013) shows that these functions can be approximated by linear planes by splitting the full displacement range into *displacement intervals*. A displacement interval is thus defined as a minimum, maximum (W_i^- and W_i^+) and an average (W_i) weight for the interval.

The sets, variables and parameters used in the model are introduced and explained below.

Sets:

$\mathcal{P}$	Set of ports.
$\mathcal{T}$	Set of ballast tanks.
$\mathcal{B}$	Set of blocks.
$\mathcal{I}$	Set of displacement intervals.
$\mathcal{BS}$	Set of bonjean stations.
$\mathcal{F}$	Set of frames.

Decision Variables:

w_{bp}	$\in \mathbb{R}^+$	Weight stowed in block $b \in \mathcal{B}$ at port $p \in \mathcal{P}$.
x_{tp}	$\in \mathbb{R}^+$	Weight of the tank $t \in \mathcal{T}$ at port $p \in \mathcal{P}$.

Auxiliary Variables:

v_p^W	$\in \mathbb{R}^+$	Vessel Displacement at port $p \in \mathcal{P}$.
v_{ip}^W	$\in \mathbb{R}^+$	Vessel Displacement in interval $i \in \mathcal{I}$ at port $p \in \mathcal{P}$.
v_{ip}^L	$\in \mathbb{R}$	Longitudinal centre of gravity at displacement interval $i \in \mathcal{I}$ at port $p \in \mathcal{P}$.
ψ_{ip}	$\in \{0,1\}$	Indicator variable for displacement interval $i \in \mathcal{I}$ at port $p \in \mathcal{P}$.
v_p^V	$\in \mathbb{R}^+$	Vertical centre of gravity at port $p \in \mathcal{P}$.
v_p^{VM}	$\in \mathbb{R}^+$	Vertical moment at port $p \in \mathcal{P}$.
v_p^M	$\in \mathbb{R}^+$	Metacentre at port $p \in \mathcal{P}$.
v_{bsp}^{Bs}	$\in \mathbb{R}^+$	Buoyancy force of section between bonjean station bs and $bs+1$ at port p.
$v_{f\alpha p}^S$	$\in \mathbb{R}^+$	Shear force fore or aft of frame $f \in \mathcal{F}$ at port $p \in \mathcal{P}$.
$v_{f\alpha p}^B$	$\in \mathbb{R}^+$	Bending moment fore or aft of frame $f \in \mathcal{F}$ at port $p \in \mathcal{P}$.

Parameters:

Various parameters:

W^O	Weight of the empty vessel.
Min^{GM}	Lower bound for metacentric height
Max_p^D	Maximum draft allowed at port p

Displacement intervals parameters:

$W_i^{\{-,+\}}$	Lower (-) and upper (+) bound of displacement interval $i \in \mathcal{I}$
W_i	Average weight of displacement interval $i \in \mathcal{I}$
$A^W_{\{M,T,D,Bs\}}(W_i)$	Weight coefficient of displacement interval $i \in \mathcal{I}$ for the linearization of metacentre (M), trim (T), draft (D), and bonjean at station bs (Bs).
$A^L_{\{M,T,Bs\}}(W_i)$	Lcg coefficient of displacement interval $i \in \mathcal{I}$ for the linearization of metacentre (M), trim (T), and bonjean at station bs (Bs).
$A_{\{M,T,D,Bs\}}(W_i)$	Constant of displacement interval $i \in \mathcal{I}$ for the linearization of metacentre (M), trim (T), draft (D), and bonjean at station bs (Bs).

Centre of gravity parameters:

$\text{Min}_i^L/\text{Max}_i^L$	Min/maximum longitudinal centre of gravity at displacement interval $i \in \mathcal{I}$
$D_b^{\{L,V\}}$	Longitudinal (L), and Vertical (V) centre of gravity of block $b \in \mathcal{B}$
$D_t^{\{L,V,T\}}$	Longitudinal (L), Vertical (V), and transversal (T) centre of gravity of ballast tank $t \in \mathcal{T}$
Max^V	Maximum vertical moment possible for the vessel.
LM^O	Longitudinal moment of the empty vessel including constant weights.
VM^O	Vertical moment of the empty vessel including constant weights.
TM^O	Transversal moment of the empty vessel including constant weights.

Bending/Shearing parameters:

$W^S_{f\alpha}$	Constant weights fore or aft of frame $f \in \mathcal{F}$
$G^\alpha_{\{b,t,bs\}f}$	Fraction of block b, ballast tank t, and buoyancy section between bonjean stations bs and $bs+1$ that lies fore or aft frame f
$W^B_{f\alpha}$	Bending components of the constant weight fore or aft of frame $f \in \mathcal{F}$
D_b^{Bs}	Distance in meters between bonjean stations bs and $bs+1$ multiplied by the density of water
$A^\alpha_{\{b,\ t,\ bs\}f}$	Fore or aft distance from frame f to the longitudinal centre of gravity of block b, ballast tank t, buoyancy section between bonjean stations bs and $bs+1$
G_f	Fore-based fraction of frame $f \in \mathcal{F}$, where $G_f \in [0;1]$. $G_f = 1$ when f is is the first frame at the bow, and $G_f = 0$ when f is the first frame at the stern.
$\text{Min}_f^{\{S,B\}}$	Lower bound for shear force (S) and bending moment (B) at frame $f \in \mathcal{F}$
$\text{Max}_f^{\{S,B\}}$	Upper bound for shear force (S) and bending moment (B) at frame $f \in \mathcal{F}$

With these the stability constraints can be modelled as seen below.

$$\sum_{t\in\mathcal{T}} x_{tp} + \sum_{b\in\mathcal{B}} w_{bp} + W^O = v_p^W \qquad \forall p\in\mathcal{P} \qquad (2.24)$$

$$\sum_{i\in\mathcal{I}} W_i^- \psi_{ip} \le v_p^W \le \sum_{i\in\mathcal{I}} W_i^+ \psi_{ip} \qquad \forall p\in\mathcal{P} \qquad (2.25)$$

$$\sum_{i\in\mathcal{I}} \psi_{ip} = 1 \qquad \forall p\in\mathcal{P} \qquad (2.26)$$

$$\sum_{i\in\mathcal{I}} v_{ip}^W = v_p^W \qquad \forall p\in\mathcal{P} \qquad (2.27)$$

$$W_i^- \psi_{ip} \le v_{ip}^W \le W_i^+ \psi_{ip} \qquad \forall i\in\mathcal{I}, p\in\mathcal{P} \qquad (2.28)$$

$$\mathrm{Min}_i^L \psi_{ip} \le v_{ip}^L \le \mathrm{Max}_i^L \psi_{ip} \qquad \forall i\in\mathcal{I}, p\in\mathcal{P} \qquad (2.29)$$

$$\sum_{b\in\mathcal{B}} D_b^L w_{bp} + \sum_{t\in\mathcal{T}} D_t^L x_{tp} + LM^O = \sum_{i\in\mathcal{I}} W_i v_{ip}^L \qquad \forall p\in\mathcal{P} \qquad (2.30)$$

$$v_p^V W_i + (1-\psi_{ip})\mathrm{Max}^V \ge v_p^{VM} \qquad \forall i\in\mathcal{I}, p\in\mathcal{P} \qquad (2.31)$$

$$v_p^V W_i - (1-\psi_{ip})\mathrm{Max}^V \le v_p^{VM} \qquad \forall i\in\mathcal{I}, p\in\mathcal{P} \qquad (2.32)$$

$$\sum_{b\in\mathcal{B}} D_b^V w_{bp} + \sum_{t\in\mathcal{T}} D_t^V x_{tp} + VM^O = v_p^{VM} \qquad \forall p\in\mathcal{P} \qquad (2.33)$$

$$\sum_{t\in\mathcal{T}} D_t^T x_{tp} + TM^O = 0 \qquad \forall p\in\mathcal{P} \qquad (2.34)$$

$$\sum_{i\in\mathcal{I}} A_M^W(W_i) v_{ip}^W + A_M^L(W_i) v_{ip}^L + A_M(W_i)\psi_{ip} = v_p^M \qquad \forall p\in\mathcal{P} \qquad (2.35)$$

$$v_p^M - v_p^V \ge \mathrm{Min}^{GM} \qquad \forall p\in\mathcal{P} \qquad (2.36)$$

$$\sum_{i\in\mathcal{I}} A_T^W(W_i) v_{ip}^W + A_T^L(W_i) v_{ip}^L + A_T(W_i)\psi_{ip} = 0 \qquad \forall p\in\mathcal{P} \qquad (2.37)$$

$$\sum_{i\in\mathcal{I}} A_D^W(W_i) v_{ip}^W + A_D(W_i)\psi_{ip} \le \mathrm{Max}_p^D \qquad \forall p\in\mathcal{P} \qquad (2.38)$$

$$\sum_{s\in\{bs,bs+1\}} \sum_{i\in\mathcal{I}} A_{Bs}^W(W_i) v_{ip}^W + A_{Bs}^L(W_i) v_{ip}^L + A_{Bs}(W_i)\psi_{ip} = 2D_d^{Bs} v_{bsp}^{Bs} \qquad \forall bs\in\mathcal{BS}, p\in\mathcal{P} \qquad (2.39)$$

$$W_{f\alpha}^S + \sum_{b\in\mathcal{B}} G_{bf}^\alpha w_{bp} + \sum_{t\in\mathcal{T}} G_{tf}^\alpha x_{tp} - \sum_{bs\in\mathcal{BS}} G_{bsf}^\alpha v_{bsp}^{Bs} = v_{f\alpha p}^S \qquad \forall f\in\mathcal{F}, \alpha\in\{A,F\}, p\in\mathcal{P} \qquad (2.40)$$

$$W_{f\alpha}^B + \sum_{b\in\mathcal{B}} A_{bf}^\alpha G_{bf}^\alpha w_{bp} + \sum_{t\in\mathcal{T}} A_{tf}^\alpha G_{tf}^\alpha x_{tp} - \sum_{bs\in\mathcal{BS}} A_{bsf}^\alpha G_{bsf}^\alpha v_{bsp}^B = v_{f\alpha p}^B \qquad \forall f\in\mathcal{F}, \alpha\in\{A,F\}, p\in\mathcal{P} \qquad (2.41)$$

$$\mathrm{Min}_f^S \le G_f v_{f,Fore,p}^S + (1-G_f) v_{f,Aft,p}^S \le \mathrm{Max}_f^S \qquad \forall f\in\mathcal{F}, p\in\mathcal{P} \qquad (2.42)$$

$$\mathrm{Min}_f^B \le G_f v_{f,Fore,p}^B + (1-G_f) v_{f,Aft,p}^B \le \mathrm{Max}_f^B \qquad \forall f\in\mathcal{F}, p\in\mathcal{P} \qquad (2.43)$$

Constraint (2.24) calculates the displacement of the vessel for every port. The next constraint, (2.25), sets the displacement interval variables together with (2.26) which ensures that exactly

one displacement interval is active at each port. Constraint (2.27) and (2.28) defines v_{ip}^W to be equal to the displacement for the active displacement interval, and 0 for the rest. In a similar fashion (2.29) and (2.30) calculates the longitudinal centre of gravity (LCG). The centre of gravity is calculated as the sum of moments divided by the total displacement. The left-hand side of (2.30) calculates the sum of moments. This is done by considering the longitudinal centre of gravity for the tanks and blocks and multiplying with the weight stowed in these. The right-hand side uses the average weight of the displacement interval, instead of the actual displacement. Similar with v_{ip}^W v_{ip}^L is zero for the displacement intervals for which $\psi_i = 0$ and for the active displacement interval it lies within the bounds defined by constraint (2.29). Constraint (2.31)-(2.33) approximates the vertical centre of gravity (VCG). Constraint (2.31) and (2.32) defines bound for the vertical moment for each displacement interval, and in the case when $\psi_i = 1$ the two inequalities turn into an equality $v_p^V W_i = v_p^{VM}$. For the non-active displacement intervals, these two constraints have no effect. Constraint (2.33) calculates the vertical moment similar to how the LCG is calculated, but without multiplying the right-hand side with the weight. (2.34) ensures that the transversal centre of gravity is 0, meaning the middle of the vessel. Due to the construction of the blocks, the transversal centre of gravity is 0 for all blocks, and thus only the tanks and the moment of the empty vessel are considered in the calculation. Constraint (2.35)-(2.39) calculates the metacentre, trim, draft and buoyancy force using the linearization of the non-linear functions. Each of the planes for the functions is described using three factors, $A^W(W_i)$, $A^L(W_i)$ and $A(W_i)$. $A^W(W_i)$ is the displacement factor, $A^L(W_i)$ is the LCG factor, and $A(W_i)$ is the constant factor. (2.35) calculates the metacentre, and (2.36) defines the metacentric height to be greater than the minimum metacentric height allowed. In (2.37) the trim is required to be zero, and (2.38) enforces the draft be less than or equal to Max_p^D. As the trim is required to be zero, the draft does not depend on the LCG, but only the displacement of the vessel. Max_p^D is the minimum draft allowed when leaving port p, and will thus be the minimum of the draft at port p and port $p+1$. Constraint (2.39) calculates the buoyancy force (*bonjean*) between station bs and $bs+1$. The last four constraints (2.40)-(2.43) are related to the stress forces. The first two calculates the shearing and bending, and (2.42)-(2.43) defines the upper and lower bounds. The shear force on a vessel, at a given frame, is the integral of forces on either side of the frame, and the bending moment is the integral of moments on either side of the frame. The buoyancy forces are only approximated, and thus there is an accumulation of error when calculating the shear force and the bending moment. To reduce the impact of this error, constraint (2.40) and (2.41) respectively calculates the shear forces and bending moment with respect to the resulting forces acting fore and aft of the frame. Hence there are two shear variables for every frame at each port. Constraint (2.42) and (2.43) respectively sets the limits for the shear force and bending moment at each frame. The shear force and bending moment at a frame are estimated as a proportional calculation based on the position of the frame. This reduces the impact of the error accumulation as the fore-based computation is accurate in the bow and the aft-based computation is accurate in the stern. All these constraints ensure that the vessel is stable and can be declared seaworthy if the stacking rules are obeyed.

The set of constraints (2.24)-(2.43) defines a polyhedron with the feasible weight allocations. This polyhedron is denoted by $\mathcal{W}$ in the model in Section 2.4.

Chapter 3

Modelling and solving the Stochastic Cargo Mix Problem

Jonas M. Christensen[a] · Alan Erera[b] · Dario Pacino[a]

[a]Management Science, Department of Management Engineering, Technical University of Denmark, Produktionstorvet, Building 424, DK-2800 Kgs. Lyngby, Denmark

[b]Georgia Institute of Technology, H. Milton Stewart School of Industrial and Systems Engineering, Atlanta, U.S.A

Publication Status: Soon to be submitted to Transportation Research: Part E

Abstract: The cargo-mix problem aims at analysing the cargo composition needed for a liner vessel to maximise its revenue on a given service. In this paper, we capture the unreliability wrt. the demand forecast in the industry by considering the cargo-flows as being stochastic instead of deterministic. To get an accurate estimation, draft, stability and capacity constraints are also included. A compact formulation of the problem is shown to be too complex to solve industrially sized instances. Instead, a rolling horizon matheuristic is presented, and the computational results show that it can achieve high-quality results in reasonable time.

Keywords: Liner shipping · Cargo composition · Maritime logistic · Block stowage · Stowage planning · Matheuristic · Stochastic Optimisation

3.1 Introduction

Aside from a few years of financial crisis, the liner shipping industry has had a continuous growth. The growing demand has resulted in a fierce competition to deliver the best product concerning efficiency, reliability, but most importantly cost. As a result, shipping rates are historically low, making it vital for the carriers to utilise their vessels as efficient as possible. In recent years, carriers have been building bigger and bigger vessels to follow demand trends, but also to achieve economies of scale.

While academic focus on vessel intake maximisation is relatively new, it is not new in the shipping industry. Container vessels are delivered with a theoretical nominal capacity. Only if the weight distribution is perfect can the full nominal capacity be reached, which hardly ever happens. With the increasing size of the vessels, a small decrease in utilisation results in a few hundred containers having to be dropped. Stowage coordinators are responsible for planning the cargo and finding a load configuration (*stowage plan*) that suits the cargo to be loaded at the current port, while also making sure the vessel can be utilised to its maximum in future ports. The unreliability wrt. the demand forecast in the industry, makes this problem even harder. For the shippers, there is no fee for booking shipments, and they will only pay for a container transport once it is undertaken by the liner. Thus, a booking does not mean the containers will ever arrive in time. Therefore, it is hard for the stowage coordinators to make a good plan as the high unreliability is not likely to change without radically changing the cost structure.

The focus of our work is the analysis of vessels' cargo mix (*the cargo mix problem*), in particular finding the cargo composition needed for a vessel to maximise its revenue on a given service. We include the unreliability of the demand, and model confirmed bookings as well as the stochasticity in demand. As for the horizon, we consider a few ports and determines the best cargo-mix for these ports. Doing so we consider cargo loaded in one of these ports, but to be unloaded at a later port. This corresponds to optimising the revenue on an interregional leg, while still considering the revenue obtained by shipping containers within the region. Delgado (2013) shows that a cargo-mix analysis based on simple capacity constraints overestimates the revenue of the vessel. Therefore it is important to include additional features and limits to estimate the capacity or revenue correctly. Additionally, we enforce that the stowage plan adheres to a block stowage strategy used within the industry. This corresponds to a logical partitioning of the vessel into blocks, and by enforcing the block strategy, each block is only allowed to host containers that have the same discharge port. This way of stowing containers is aimed at improving operations at ports since it makes it possible to perform, e.g. dual cycling (where load and discharge operations are no longer sequential). Examples of block stowage are found in the stowage planning literature in e.g. Ambrosino et al. (2015b) and Ambrosino et al. (2015a).

The proposed model can have multiple applications, e.g., driving rate prices, improving fleet composition and network design. However we see greater potential applying the model as an analytical tool, and we envision it will mostly be used to perform different what-if analyses. To allow for multiple what-if analyses to be performed in a short period, the solution method needs to be fast, and we will put a lower priority of the accuracy wrt. optimality and prioritise an adaptable, simple and fast solution method.

In Christensen and Pacino (2017) a deterministic version of the cargo mix problem is described,

and multiple matheuristics based on the same idea are compared. This paper can be seen as a stochastic extension of the method developed in Christensen and Pacino (2017), and thus contributes to the state-of-the-art by first extending the formal definition of the cargo mix problem from Christensen and Pacino (2017) to include stochastic cargo flows and accepted bookings. Second, we include the stochasticity in the compact formulation of Christensen and Pacino (2017). Lastly, we show that the compact formulation can only solve the smallest of the instances using standard mixed-integer programming methods, and therefore we propose a matheuristic to solve the problem.

The remainder of the paper is organised as follows. Section 3.2 present background knowledge of vessel architecture and the industry as a whole. In Section 3.3, relevant existing literature is reviewed. Section 3.4 gives a detailed description of the problem and presents a compact formulation for the problem. Section 3.5 describes the matheuristic approach. Section 3.6 describes the data generated and used for this study, and Section 3.7 presents the results for the matheuristic and compares it with the result for the compact model. Lastly, Section 3.8 contains the final remarks and conclusions.

3.2 Background

Liner shipping is the service of long-haul transportation of goods by using ocean-going vessels. The vessels used are of high capacity and operate on a fixed route with published schedules. The high capacity of the vessels helps to keep costs down, and the liner shipping industry is the cheapest and most energy-efficient form of international transportation.

The cargo to be transported is packed in standardised containers, which are then are loaded on vessels. Most containers carried on liner vessels are 8 feet wide, 8'6" high and *20'*, *40'* or *45'* long. *High-cube* containers also exist, which are 9'6" high. Perishable goods are packed in so-called *reefer* containers. These are refrigerated containers with an integrated cooling unit to keep the cargo cool. The cooling unit must get power from the vessel, and thus reefer containers can only be stowed where an electrical outlet is available. Additionally, there are dangerous goods containers, which must adhere to a certain safety standard describing the segregation rules for different kind of dangerous material (e.g. toxic, nuclear or flammable).

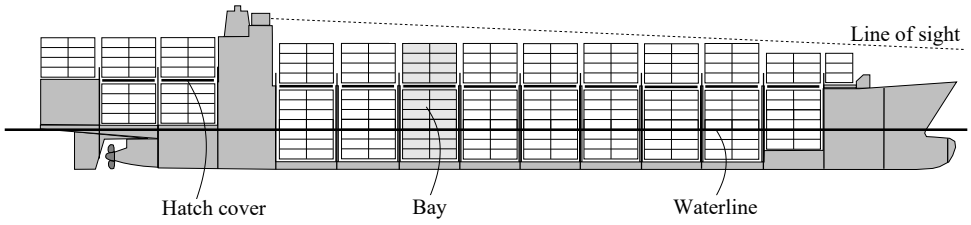
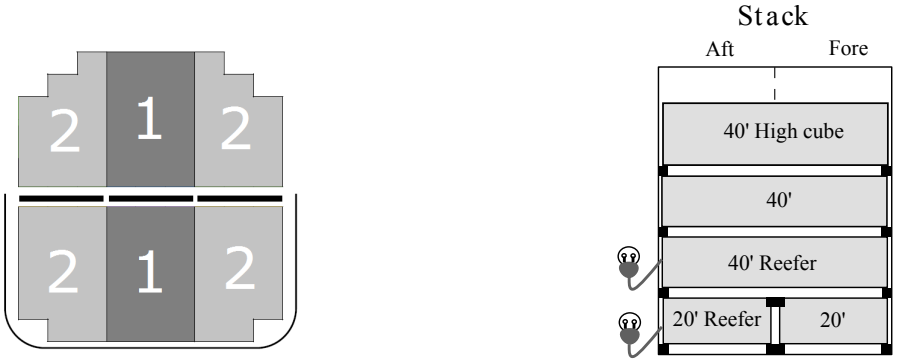


Figure 3.1: Example figure of a container vessel.

Figure 3.1 shows the layout of a vessel. A vessel is divided into *bays*, and for each bay there is an *on-deck* and *below-deck* part, which are physically separated by a *hatch cover*. Hatch covers are

flat and leak-proof structures that can be removed when loading and unloading containers. The bays are further divided into a number of *stacks* and *cells*. A cell can hold a single 40' container or two 20' containers, and the stack refers to a cell's longitudinal/transversal position on the vessel. Figure 3.2 outlines the design of a bay. First, Figure 3.2a shows a transversal section of a bay, where the numbers denote the blocks. Secondly, Figure 3.2b shows the layout of a stack within a bay.



(a) A bay in the vessel *(b) The layout of a stack*

Figure 3.2: Example figure showing the design of a bay, and a stack.

A cell consists of two *slots*; an *aft* (towards the stern) and a *fore* (towards the bow) slot. For each stack, there are weight and height limits on the containers stowed. Furthermore, there are specific stacking rules, describing how the containers can be stowed. E.g., 20' containers cannot be stowed above 40' containers, and generally, the weight of the containers must decrease upwards in the stacks on deck.

The main cost components in liner shipping are the fuel costs and the costs associated with port stays. The costs at port depend on the time spent at the port, and thus decreasing the time spent at the port will most often decrease the associated costs as well. To reduce the time at port, it is important to avoid *overstowage*, and especially *hatch overstowage*. Overstowage occurs when a container is stowed below a container destined for a later port. Hence, to unload the desired container, you will need to move the overstowing container first. Hatch overstowage happens when a container on-deck is overstowing a container below deck, in which case all containers on-deck will need to be moved in order to remove the hatch cover. These excess cranes moves are costly and take time, and the number of these moves should, therefore, be minimised. The benefit of decreasing the time at port is two-fold; the port costs are minimised, and if less time is spent in port, the vessel can sail slower to the next port and thus save fuel.

The profit comes from the transported containers, and naturally, carriers want to maximise the capacity usage to increase revenue. Vessel's capacities are measured in the number of *Twenty-foot Equivalent Units* (TEUs) the vessel can carry under perfect conditions, i.e. the number of slots on the vessel. A 40' foot standard container and 40' high-cube container are 2 TEU or 1 *FEU* (*Forty-foot Equivalent Unit*).

To facilitate better planning and ease port handling operations vessels are divided into *blocks* (a virtual grouping of stacks). In the industry, different companies have different definitions of a block. In accordance with our industry collaborator, we define a block such that stacks below and above the same hatch-cover belong to the same block. When stowing the ship, all containers

in a block must have the same discharge port. Therefore overstowage and hatch-overstowage cannot happen.

Before a vessel leaves port, the captain of the ship must deem the vessel *seaworthy*. Besides ensuring that the vessel is not defective or undermanned, this also entails that the static stability is correct and all stress forces are within limits. The weight of the cargo causes shearing and bending stresses over the vessel structure, which must be within limit at certain calculation points. Ensuring the vessel is stable also requires the three coordinates of the centre of gravity (longitudinal, vertical and transversal) to be within defined limits. The *aft draft* is the distance between the waterline and the bottom of the hull at the stern of the ship, and the *fore draft* is at the bow. The trim is defined as the difference between the aft draft and the fore draft. The trim, fore draft, and aft draft must all be within limits.

The outline of a vessel is described in a document called the *vessel profile*, which specifies all the weight and capacity limits along with the *hydrostatic table*. Using the hydrostatic table one can calculate the draft, centre of gravity and metacentre. For a more detailed description of the stability calculation and requirements see Delgado (2013).

3.3 Literature Review

The existing literature on the liner shipping cargo mix problem is limited. The problem was formally introduced in the PhD thesis of Delgado (2013). Here a mixed integer programming model was presented, and the multi-port version is shown not to be scalable. To achieve scalability, the problem is decomposed in a similar way as to what is suggested in earlier stowage planning work (see Pacino et al. (2011)). The work of Christensen and Pacino (2017) extends the liner shipping cargo mix problem, by including the concept of block stowage, draft restrictions and restricts the number of containers that can be selected. The inclusion of the block stowage strategy makes the MIP model proposed in Delgado (2013) intractable for industrially sized instances. Instead, a novel matheuristic is developed. Tested on industrially sized instances, the matheuristic is shown to be scalable and produce high-quality solutions in a matter of seconds. A revenue model for the short-sea shipping service is presented in Feng and Chang (2008). This model disregards most aspects of what it means for a vessel to be seaworthy, and only considers the TEU and weight capacity. In Delgado (2013) it is shown that, due to inaccuracies wrt. stability considerations, the model in average overestimates the revenue by 8%. To get an accurate estimate on the revenue, we, therefore, believe it is important to consider the seaworthiness of the vessel with care. Our work differs from previous work by also including the unreliability wrt. the demand forecast within the industry. This is done by considering the cargo flow as being stochastic instead of deterministic. Doing so, a list of ports are considered instead of a circular service, and the vessel is assumed to already be loaded with some containers.

The cargo-mix problem can be seen as a generalisation of a stowage planning problem. Given the limited amount of existing literature on the cargo mix problem, it is therefore relevant to introduce relevant literature related to stowage planning. The main difference between the cargo mix problem and stowage planning is that the cargo mix problem considers a set (or in this case; distribution) of expected containers that is possible to be loaded in the visited ports. Whereas in stowage planning a list of containers has already been selected, and the main decision is where

these containers should be stowed on the vessel. In the stowage planning problem literature, the seaworthiness of the vessels is being considered with varying degree of importance. Botter and Brinati (1992) presents an accurate formulation which includes limits on the metacentric height, transversal stability, trim, shear forces and bending moments. The model fails to solve, and instead, a heuristic decomposition method is used. The approach by Pacino et al. (2011) considers metacentric height, trim and shear forces as well as draft, and is in Pacino et al. (2012) extended to also include direct linearization of hydrostatic data and the modelling of ballast tanks.

Integer programming models assigning containers to specific slots on the vessel are presented in Botter and Brinati (1992), Ambrosino et al. (2004), and Li et al. (2008). All these models experience scalability issues, due to a large number of variables and constraints. Scalable integer programming approaches (Wilson and Roach, 2000; Kang and Kim, 2002; Ambrosino et al., 2010; Pacino et al., 2011) all rely on a multiphase approach. In these approaches, the overall problem is decomposed into a master planning phase and a slot planning phase. The master planning phase determines which general area a container should be stowed in following high-level capacity and stability constraints. In the slot planning phase, individual containers are assigned a specific slot. Delgado (2013) shows that the master planning phase can be used to approximate the revenue accurately.

In Christensen and Pacino (2017) it is shown that the deterministic cargo mix problem is too complex to be solved using standard mixed integer programming methods. Based on these results we do not expect to be able to solve the stochastic version to optimality within a time that makes it applicable to the industry. It is therefore relevant to look at specialised heuristic methods for stochastic problems.

Large stochastic MIPs are computationally hard to solve due to a large number of scenarios. One way to handle this is to use a rolling horizon heuristic (RHH). Such a planning procedure only considers a portion of the entire planning horizon at a time, solves this reduced problem and fixes parts of the solution. Rolling horizons schemes have been applied to several problems within operations management (see Chand et al. (2002) for an extensive review). Due to the myopic nature of the procedure, it is highly appropriate when only limited information is available. However, it can also be used to reduce the problem size, and thereby lessen the computational effort to solve the problem.

The literature on rolling horizon heuristics within a maritime setting is limited. Rakke et al. (2011) describes a rolling horizon heuristic for a maritime routing and inventory management problem. A reduced version of the mathematical model is used as an improvement heuristic where the feasible solution found by the RHH is used to reduce the number of variables. The results show that the RHH outperforms the MIP model, and produces quality solutions in a relatively short amount of time. Additionally, the improvement heuristic is shown to improve the result by $2\% - 10\%$ for the datasets tested. Bredstrom and Rönnqvist (2006) applies a rolling horizon heuristic to a combined supply chain and ship routing problem. Real-world industrially sized data is used, and the heuristic is shown to outperform the MIP model.

3.4 The Stochastic Cargo Mix Problem with Block Stowage (SCMPBS)

Given a vessel, a string of ports, an initial configuration of the vessel, a list of accepted bookings and a distribution of the cargo flow (i.e. origin-destination demand matrix), the Stochastic Cargo Mix Problem with Block Stowage (SCMPBS) aims at optimising the expected revenue of the cargo loaded. A high degree of accuracy is imposed wrt. stability constraints, to ensure the vessel is seaworthy. Furthermore, the loading must obey the block stowage requirement, ensuring that all containers in a block must have the same discharge port. We will not assume the vessel is initially empty, and the initial configuration of the vessel will describe the cargo already loaded, as well as their destination.

The string of ports is divided in two; a set of visited ports, for which unload and loading operations will be planned, and a set of demand ports for which only unload operations will be scheduled. This separation corresponds to a situation where a vessel is in the Far East, and the carrier wants to optimise the revenue of the containers it brings to Europe. When the vessel arrives in Europe, the demand is known more accurately, and the problem can be solved once again taking the current vessel configuration into consideration.

The aim is to determine the optimal cargo-mix, and not make a fully feasible stowage plan. In Pacino et al. (2011) it is shown that working on an aggregated level gives a good approximation to the full problem. Therefore, the problem is decomposed similarly to earlier work (Wilson and Roach, 2000; Kang and Kim, 2002; Ambrosino et al., 2010; Pacino et al., 2011; Delgado, 2013), which also helps to ease the computational effort needed to solve the problem. Instead of assigning specific containers to specific slots on the vessel, containers are grouped in *container types* and assigned to *blocks*. Each container type represents a number of containers with the same properties wrt. weight, height, length and reefer capabilities. The blocks correspond to a logical partitioning of the vessel into non-overlapping sections. In the decomposition, the planning is split in two; first, a master planning phase where container types are assigned to blocks while satisfying high-level capacity constraints. Hereafter, a slot planning phase, consider each block sequentially and assign a specific slot for each of the containers assigned in the master planning phase. This decomposition is illustrated in Figure 3.3. Delgado (2013) showed that the master planning phase gives an accurate estimation of the achievable capacity utilisation, as this is the goal of this study, we disregard the slot planning phase.

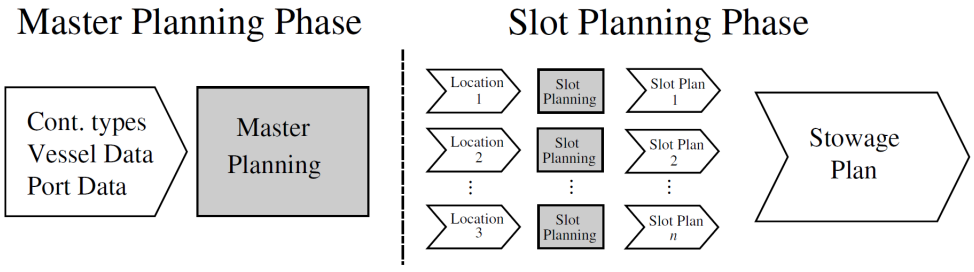


Figure 3.3: The two-phase heuristic decomposition

For the container types, we consider the most standard containers: 20' dry and reefer containers, 40' dry and reefer containers both normal height and high-cube. All with a set of weight classes,

ranging from 3 tonnes (empty) to 27 tonnes (full).

The described problem can be modelled as a multi-stage stochastic programming problem. The decision process is thus as follows;

Observe demand at port 1 $\rightarrow$ Decide block assignment for port 1 $\rightarrow$ Find containers to load at port 1 $\rightarrow$
Observe demand at port 2 $\rightarrow$ Decide block assignment for port 2 $\rightarrow$ Find containers to load at port 2 $\rightarrow \ldots \rightarrow$
Observe demand at port n $\rightarrow$ Decide block assignment for port n $\rightarrow$ Find containers to load at port n

When determining the block assignment at port p, only the blocks being emptied at port p will need a new destination port assigned. The rest of the blocks will have to keep the destination that has already been assigned, in order not to move the containers. The decision process can be illustrated by a scenario tree, as shown in Figure 3.4, where Low, Medium and High describes three possible realisations of the demand at port a. A scenario is defined as a path from the root of the tree to one of the leaves. The nodes visited by each path corresponds to a realisation of the random parameters in the model. Let $\mathcal{S}$ be the set of scenarios, and $\mathcal{N}$ the set of nodes in the scenario tree, each one of these nodes $n \in \mathcal{N}$ corresponds to a vector of random parameters with a particular history up to that node.

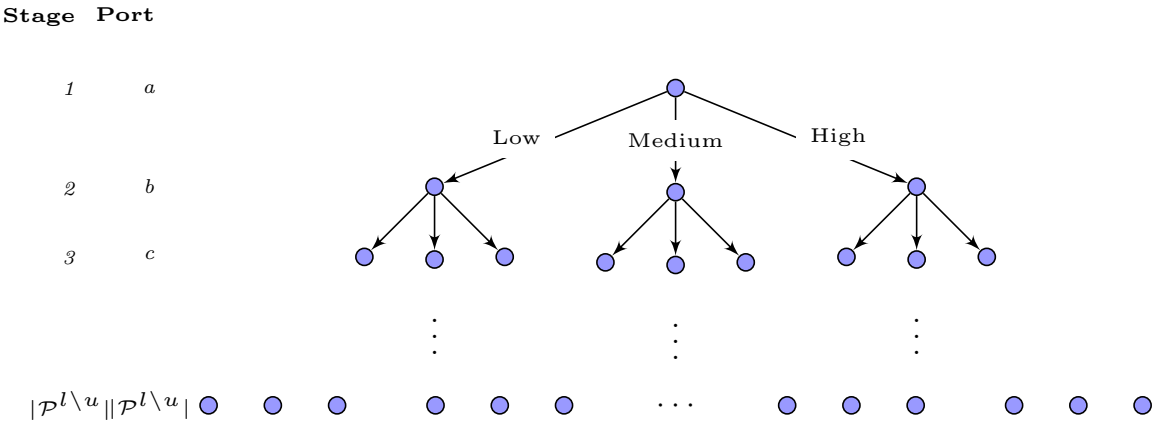


Figure 3.4: Scenario tree illustrating the multi-stage decision process.

Let $\pi_{nn'}$ be the transitional conditional probability of going from node n to node n', and furthermore let Π_n be the unconditional probability of node n. Π_n is computed by multiplying the conditional probabilities of nodes positioned on the path starting at the root node and ending at node n.

$$\Pi_n = \prod_{\langle n_1, n_2 \rangle \in path(n)} \pi_{n_1, n_2}$$

Here $path(n)$ describe the path from the root to node n. The scenario probabilities Π_s can be computed similarly.

Along with the sets $\mathcal{S}$ and $\mathcal{N}$, define $\mathcal{C}$ as the set of container types. A container type, $c \in \mathcal{C}$, is defined by the dimensions and properties of the containers. The parameters Γ^c, Φ^c, v^c describes respectively the TEU coefficient (1 for a 20' container, 2 for a 40' container), volume and weight of container type $c \in \mathcal{C}$. The volume is needed for the modelling of the high-cube containers,

which need further restrictions than a simple capacity constraint. Also, let the set $\mathcal{B}$ be the set of blocks. Let $\mathcal{P}$ be the full set of ports, $\mathcal{P} = \mathcal{P}^{l\setminus u} \cup \mathcal{P}^{u}$ where $\mathcal{P}^{l\setminus u}$ is the set for which both loading and unloading operations will be planned, and $\mathcal{P}^{u}$ the set of ports where only unloading operations are planned. The container demand is described by the set $\mathcal{T}$ and the parameter a_n^{dc}. The set $\mathcal{T}$ is the set of transports. A transport describes an origin-destination pair and a_n^{dc} is the number of containers available of type $c \in \mathcal{C}$ during transport $t = \langle p^n, d \rangle$ in node $n \in \mathcal{N}$. Here p^n is the port associated with node $n \in \mathcal{N}$. The set $\mathcal{T}_p^{ON}$ is the set of transports where port p is visited between the origin and destination of the transport, defined as follows.

$$\mathcal{T}_p^{ON} = \{t \in \mathcal{T} |\, o^t \leq p < d^t\} \qquad \forall p \in \mathcal{P}$$

Here o^t and d^t are the origin and destination port of transport t.

The main decision variable is $y_{bs}^{tc} \in \mathbb{R}^+$, defining the (possibly fractional) number of containers of type $c \in \mathcal{C}$ of transport $t \in \mathcal{T}$ to be stowed in block $b \in \mathcal{B}$ in scenario $s \in \mathcal{S}$. Naturally the container load variables must be an integer, however, as earlier mentioned this work mostly concerns revenue estimation, and the aim is thus not to generate a fully feasible stowage plan. Christensen and Pacino (2017) shows that relaxing the integrality requirement of the container load variables gives an accurate estimation of the revenue, and furthermore describes how the resulting solution can be converted into a solution with integer container load variables.

Let $w_{bps} \in \mathbb{R}^+$ be a variable denoting the weight stowed in block $b \in \mathcal{B}$ at port $p \in \mathcal{P}^{l\setminus u}$ in scenario $s \in \mathcal{S}$. To enforce the block stowage requirement, we introduce the variable σ_{bn}^d, defined as follows

$$\sigma_{bn}^d = \begin{cases} 1 & \text{If block } b \in \mathcal{B} \text{ is assigned destination } d \in \mathcal{P} \text{ in node } n \in \mathcal{N} \\ 0 & \text{Otherwise.} \end{cases}$$

Let u^{tc} describe the accepted bookings i.e. the number of accepted containers of type $c \in \mathcal{C}$ for transport $t \in \mathcal{T}$. From the initial configuration of the vessel we define two parameters, θ_{bp}^c and Ω_{bp}^d. θ_{bp}^c describes the cargo loaded i.e number of containers of type $c \in \mathcal{C}$ initially loaded that still occupy block $b \in \mathcal{B}$ at port $p \in \mathcal{P}$, whereas Ω_{bp}^d describe the partial block assignment imposed by the initial configuration. That is;

$$\Omega_{bp}^d = \begin{cases} 1 & \text{If the initial configuration imposes that block } b \in \mathcal{B} \text{ is assigned} \\ & \text{destination } d \in \mathcal{P} \text{ at port } p \in \mathcal{P}^{l\setminus u} \\ 0 & \text{Otherwise.} \end{cases}$$

Let $\mathbf{y}_s \in \mathbb{N}^{|\mathcal{B}|\cdot|\mathcal{T}|\cdot|\mathcal{C}|}$ be the vector of y_{bs}^{tc} variables for scenario $s \in \mathcal{S}$. Then define the polyhedron $\mathcal{Y}$ as the loading configurations that satisfy the capacity constraints for the variables $\mathbf{y}_s$.

$$\mathcal{Y} = \left\{ s \in \mathcal{S}, \mathbf{y}_s \in \mathbb{R}^{|\mathcal{B}|\cdot|\mathcal{T}|\cdot|\mathcal{C}|} \;\; s.t \;\; (3.35) - (3.39) \right\}$$

In essence, the vector $\mathbf{y}_s$ describe a full solution for a scenario, and the polyhedron $\mathcal{Y}$ is a set of constraints ensuring the solution is feasible wrt. the capacity constraints. Similarly, define $\mathbf{w}_s \in \mathbb{R}^{|\mathcal{B}|\cdot|\mathcal{P}|}$ as the vector of w_{bps} variables for scenario $s \in \mathcal{S}$. Then define the polyhedron $\mathcal{W}$ as follows

$$\mathcal{W} = \left\{ s \in \mathcal{S}, \mathbf{w}_s \in \mathbb{R}^{|\mathcal{B}|\cdot|\mathcal{P}|} \;\; s.t \;\; (3.40) - (3.59) \right\}$$

Here (3.40) - (3.59) describe the feasible weight allocations that ensures the seaworthiness of the vessel. Where the polyhedron $\mathcal{Y}$ describe the feasibility regarding capacity, the polyhedron $\mathcal{W}$ describe the feasibility concerning the seaworthiness of the vessel. The polyhedron $\mathcal{Y}$ is formally described in Section 3.A, and the stability constraints are described in Christensen and Pacino (2017). For the reader's convenience the description of the stability constraints are attached here in Section 3.B.

Below, all the sets, variables and parameters are summarized, and additional sets and parameters are introduced.

Sets:

$\mathcal{S}$	Set of scenarios
$\mathcal{N}$	Set of nodes
$\mathcal{N}(s,o,d)$	Set of nodes for scenario s between port o and d (including o, excluding d) $\mathcal{N}(s,o,d) = \{n \in \mathcal{N} \vert n \in s, o \leq p^n \wedge p^n < d\}$
$\mathcal{S}(n)$	Set of scenarios having node $n \in \mathcal{N}$ on the path. $(\mathcal{S}(n) = \{s \in \mathcal{S} \vert n \in s\})$
$\mathcal{P}$	Set of ports
$\mathcal{P}^{l\backslash u} \subset \mathcal{P}$	Set of load/unload ports
$\mathcal{P}^{u} \subset \mathcal{P}$	Set of demand ports (only unload operations will be planned)
$\mathcal{T}$	Set of transports
$\mathcal{T}_p^{ON}$	Set of transports that visits port $p \in \mathcal{P}$
$\mathcal{C}$	Set of container types
$\mathcal{B}$	Set of blocks.

Variables:

$y_{bs}^{tc} \in \mathbb{R}^+$	Number of containers of type $c \in \mathcal{C}$ of transport $t \in \mathcal{T}$ to be stowed in block $b \in \mathcal{B}$ in scenario $s \in \mathcal{S}$.
$w_{bps} \in \mathbb{R}^+$	Weight of the containers stowed in block $b \in \mathcal{B}$ at port $p \in \mathcal{P}^{l\backslash u}$. in scenario $s \in \mathcal{S}$.
$\sigma_{bn}^{d} \in \{0,1\}$	Variable denoting whether or not block $b \in \mathcal{B}$ can only contain containers with destination $d \in \mathcal{P}$ in node $n \in \mathcal{N}$

Parameters:

$\Pi_s \in [0;1]$	Probability for scenario $s \in \mathcal{S}$.
$f^{tc} \in \mathbb{R}^+$	The value of container type $c \in \mathcal{C}$ of transport $t \in \mathcal{T}$.
$p^n \in \mathcal{P}$	The port associated with node $n \in \mathcal{N}$.
$v^c \in \mathbb{R}^+$	Weight of container type $c \in \mathcal{C}$
$\theta_{bp}^{c} \in \mathbb{N}$	Number of containers of type $c \in \mathcal{C}$ initially loaded that still occypy block $b \in \mathcal{B}$ at port $p \in \mathcal{P}$
$\Omega_{bp}^{d} \in \{0,1\}$	Parameter denoting if block $b \in \mathcal{B}$ is enforced to only contain containers with destination $d \in \mathcal{P}$ at port $p \in \mathcal{P}^{l\backslash u}$.
$a_n^{dc} \in \mathbb{N}$	Number of available containers of type $c \in \mathcal{C}$ for transport $t = \langle p^n, d \rangle$ at node $n \in \mathcal{N}$.
$u^{tc} \in \mathbb{N}$	Number of already accepted containers of type $c \in \mathcal{C}$ for transport $t \in \mathcal{T}$.
$q_b \in \mathbb{R}^+$	Weight capacity of block $b \in \mathcal{B}$

With this we can model the problem as follows

$$\texttt{Max Z} = \sum_{s\in\mathcal{S}}\sum_{b\in\mathcal{B}}\sum_{t\in\mathcal{T}}\sum_{c\in\mathcal{C}} \Pi_s f^{tc} y_{bs}^{tc} \tag{3.1}$$

Subject to:

$$\mathbf{w}_s \in \mathcal{W} \qquad \forall s \in \mathcal{S} \tag{3.2}$$

$$\mathbf{y}_s \in \mathcal{Y} \qquad \forall s \in \mathcal{S} \tag{3.3}$$

$$y_{bs}^{tc} = y_{bs'}^{tc} \qquad \forall s \in \mathcal{S}, n \in s, s' \in \mathcal{S}(n), b \in \mathcal{B}, d \in \mathcal{P}, t = \langle p^n, d\rangle, c \in \mathcal{C} \tag{3.4}$$

$$\sum_{t\in\mathcal{T}_p^{ON}}\sum_{c\in\mathcal{C}} v^c y_{bs}^{tc} + \sum_{c\in\mathcal{C}} v^c \theta_{bp}^c \le w_{bps} \qquad \forall s \in \mathcal{S}, b \in \mathcal{B}, p \in \mathcal{P}^{l\setminus u} \tag{3.5}$$

$$\sigma_{bn}^d \ge \Omega_{bp^n}^d \qquad \forall n \in \mathcal{N}, b \in \mathcal{B}, d \in \mathcal{P} \tag{3.6}$$

$$\sum_{d\in\mathcal{P}} \sigma_{bn}^d = 1 \qquad \forall n \in \mathcal{N}, b \in \mathcal{B} \tag{3.7}$$

$$y_{bs}^{tc} \le \max\left(a_n^{dc}, u^{tc}\right)\sigma_{bn'}^d \qquad \forall n \in \mathcal{N}, s \in \mathcal{S}(n), b \in \mathcal{B}, c \in \mathcal{C}, d \in \mathcal{P}, t = \langle p^n, d\rangle, n' \in \mathcal{N}(s, p^n, d) \tag{3.8}$$

$$\sum_{b\in\mathcal{B}} y_{bs}^{tc} \le \max\left(a_n^{dc}, u^{tc}\right) \qquad \forall n \in \mathcal{N}, s \in \mathcal{S}(n), d \in \mathcal{P}, c \in \mathcal{C}, t = \langle p^n, d\rangle \tag{3.9}$$

$$\sum_{b\in\mathcal{B}} y_{bs}^{tc} \ge u^{tc} \qquad \forall s \in \mathcal{S}, t \in \mathcal{T}, c \in \mathcal{C} \tag{3.10}$$

$$0 \le w_{bps} \le q_b \qquad \forall s \in \mathcal{S}, b \in \mathcal{B}, p \in \mathcal{P}^{l\setminus u} \tag{3.11}$$

$$y_{bs}^{tc} \in \mathbb{R}^+ \qquad \forall s \in \mathcal{S}, b \in \mathcal{B}, t \in \mathcal{T}, c \in \mathcal{C} \tag{3.12}$$

$$\sigma_{bn}^d \in \{0,1\} \qquad \forall n \in \mathcal{N}, b \in \mathcal{B}, d \in \mathcal{P} \tag{3.13}$$

Objective (3.1) maximises the expected value of the containers loaded, where f^{tc} is a general function describing the value of each container. This parameter can be changed to accommodate different objectives, e.g. intake maximisation as in Christensen and Pacino (2017). Here it is the revenue.

Constraint (3.2) and (3.3) ensures that the stowage plans for all the scenarios are feasible with respect to stability and capacity respectively. The parameter θ_{bp}^c is used in the capacity constraints to reduce the capacity of the blocks, according to the initially loaded containers.

Constraint (3.4) are the non-anticipativity constraints, making sure that the decisions for two scenarios with a common history are the same up until that point. In other words; no information revealed at a later stage is utilised at the current stage. Constraint (3.5) links the weight variables w_{bps} with the weight of the containers stowed in the scenario plus the weight of the containers initially on the vessel. Intuitively the left-hand side and right-hand side of (3.5) should be equal. However, the work of Christensen and Pacino (2017) relies on relaxing this equality constraint, and later undo the relaxation, ensuring a strict correspondence between the weight variable and

the weight of the containers. It is shown that the relaxed model gives an accurate revenue estimation, in considerably less time. Our main focus is to get an accurate revenue estimation fast, and therefore (3.5) does not impose a strict correspondence for the weight variables.

Constraints (3.6)-(3.8) are the block assignment constraints. Constraint (3.6) makes sure the block assignment follows that imposed by the initial configuration. Together (3.7) and (3.8) enforce that no overstowage can occur. Specifically, constraint (3.7) ensures that exactly one port is assigned as the discharge port for every node. Constraint (3.8) makes sure that containers to a destination d can only be stowed in a block if the assigned destination matches d, during the full journey. Constraint (3.7) does not enforce that if $\sigma_{bn}^{d'} = 1$ for a block b then all nodes with $p^n < d'$ must have d' as discharge port. This will, however, be the case regardless since constraint (3.8) is posted for every node, which effectively disallows the stowage of containers with another port of destination.

Constraint (3.9) ensures that no more containers than the ones available in the scenario are loaded, and constraint (3.10) make sure we load the containers already accepted. Finally constraints (3.11)-(3.13) defines the variables domain, and (3.11) enforces the block weight capacity limit. The model includes ballast water within the set $\mathcal{W}$, but only as a way to fix stability issues, thus the ballast water is not minimised as part of the objective as in Delgado (2013).

3.5 Solution Method

The number of scenarios in the stochastic model in Section 3.4 grows exponentially with the number of ports considered. As the number of scenarios quickly increases, even more so does the number of variables. Therefore it is not expected that the model can solve more than the smallest toy example.

Besides the number of variables, the main contributor to the intractability of the stochastic model is the binary indicator variables enforcing the block stowage. To overcome this, we will describe a multiphase hierarchical heuristic where the block assignment is heuristically determined in Phase I. Hereafter a rolling horizon heuristic is used in Phase II to deal with the stowing of containers.

The solution method bears a resemblance to the solution method proposed in Christensen and Pacino (2017) in which a deterministic version of the problem is considered.

3.5.1 Phase I

Phase I heuristically determines the block assignment. In Section 3.4 the block assignment is allowed to be different from scenario to scenario. We will disregard this in this matheuristic, and just set one full block assignment a priori. The block assignment is determined for all ports $p \in \mathcal{P}^{l \setminus u}$. For all blocks, we want to find a block assignment that fit the attributes of the specific block. First, we will describe how the block assignment can be determined if no bookings have been accepted ($u^{tc} = 0$ for all transports and container types). Hereafter this method will be extended to handle accepted bookings.

3.5.1.1 No accepted containers

When no containers has been accepted, the block assignment can be determined by solving a longest path problem in a directed acyclic graph. Figure 3.5 is an example of such a graph with $|\mathcal{P}^{l\backslash u}| = 4$ and $|\mathcal{P}^{u}| = 4$. The ports in $\mathcal{P}^{l\backslash u}$ are labelled a, b, c and d, where as the ports labelled 1, 2, 3 and 4 belong to the set $\mathcal{P}^{u}$. An edge in the graph describes an assignment of a destination port to an origin port, and a s-t path in the graph G describes a block assignment, in which all ports in $\mathcal{P}^{l\backslash u}$ are assigned a destination port $d \in \mathcal{P}$. If a path uses an edge (p', d'), port d' will be assigned as the destination port for port p'. At all ports $p \in \mathcal{P}^{l\backslash u}$ in between p' and d', d' will have to be assigned as the destination port as well.

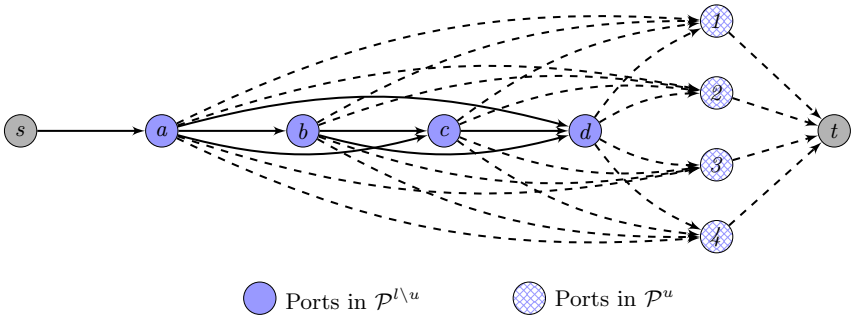


Figure 3.5: An example graph G with $|\mathcal{P}^{l\backslash u}| = |\mathcal{P}^{u}| = 4$.

Let G be a weighted directed acyclic graph with V as the vertex set, and E the edge set. The graph consist of a source s, and a sink node t. Beside these two nodes each port $p \in \mathcal{P}$ is represented as a node in the graph, such that

$$V = \{s, t\} \cup \mathcal{P}$$

To define the edge set, let E_{base} be the set of *base edges*, i.e. edges that connect the sink and source node to the rest of the graph.

$$E_{base} = (s, p_0) \cup \{(i, t) | i \in \mathcal{P}^{u}\}$$

Where p_0 is the first port in the service (a in Figure 3.5). Define $\delta^{+}(i)$ as the set of ports visited after port i in the service. With this define the full set of edges as

$$E = E_{base} \cup \{(i, j) | i \in \mathcal{P}^{l\backslash u}, j \in \delta^{+}(i)\}$$

The initial configuration implies a partial block assignment for some blocks, which will need to be satisfied when generating the full block assignment. We control this by setting the edge weight, such that the longest path in the graph ensures that the partial block assignment is fulfilled. If for a block b and port p_0 there exists a destination $d \in \mathcal{P}$ such that $\Omega^{d}_{bp_0} = 1$, then the destination at the first port is imposed by the initial configuration of the vessel. In that case, we want to make sure that port d is assigned as the destination for all ports p visited before port d. Thus we want to ensure that the longest path in the graph G contains the edge (p_0, d). This also excludes a set of edges we know cannot be used. The set of *invalid* edges for a block b is denoted as $E_I(b)$, and defined as follows.

$$E_I(b) = \{(i, j) \in E,\ d \in P \mid i < d, j \in \delta^{+}(i), \Omega^{d}_{bp_0} = 1\} \setminus (p_0, d)$$

To ensure the set of invalid edges cannot be part of the longest path we set the weight for all these edges to $-M$

$$w_{ij} = -M \qquad \forall (i,j) \in E_I(b)$$

For all valid, non-base edges ($(i,j) \in E \setminus \{E_{base} \cup E_I(b)\}$) the weight represent the maximum revenue that can be transported along this edge. The calculation considers the blocks TEU capacity, reefer capacity as well as the demand. Algorithm 2 describe how this is calculated.

Algorithm 2 `Calculate` w_{ij}

Input: G, $\overline{D}$, F, S, b
1: $E_I(b) \leftarrow$ `makeInvalidEdges`(G, S, b)
2: **for all** $(i,j) \in E \setminus \{E_{base} \cup E_I(b)\}$ **do**
3: $w_{ij} \leftarrow 0$
4: $\Delta\tau \leftarrow \max\left(k_b^{20}, k_b^{TEU}\right)$
5: $\Delta\tau^r \leftarrow \max\left(r_b^{Slot}, 2r_b^{Cell}\right)$
6: $\mathcal{T}_{ij} = \{t \in \mathcal{T} | o^t \geq i \wedge d^t = j\}$
7: $\mathcal{K} \leftarrow$ All tuples $\langle t, c, f^{tc}\rangle$ where $t \in \mathcal{T}_{ij}$, $c \in \mathcal{C}$, and f^{tc} being the value of container type $c \in \mathcal{C}$ of transport $t \in \mathcal{T}_{ij}$.
8: $\mathcal{K} \leftarrow$ Sort the set $\mathcal{K}$ after descending order of f^{tc}.
9: **for all** $\langle t, c, f^{tc}\rangle \in \mathcal{K}$ **do**
10: **if** $c \in \mathcal{R}$ **then**
11: capacity $\leftarrow \min(\Delta\tau,\ \Delta\tau^r)$
12: **else**
13: capacity $\leftarrow \Delta\tau$
14: **end if**
15: load $\leftarrow \min\left(d^{tc}, \lfloor \frac{\text{capacity}}{\Gamma^c} \rfloor\right)$
16: $w_{ij} \leftarrow w_{ij} + f^{tc}$load
17: $\Delta\tau \leftarrow \Delta\tau - \Gamma^c$load
18: **if** $c \in \mathcal{R}$ **then**
19: $\Delta\tau^r \leftarrow \Delta\tau^r - \Gamma^c$load
20: **end if**
21: **end for**
22: **end for**
23: $w_{ij} = 0 \qquad \forall (i,j) \in E_{base}$
24: $w_{ij} = -M \qquad \forall (i,j) \in E_I(b)$
25: **return** W

The input for Algorithm 2 is the following: The graph G, average demand- and value matrix $\overline{D}$ and F, the initial configuration S and the block b. The algorithm returns the edge weight matrix W to be used when calculating the block assignment for the given block. Line 1 makes the set of invalid edges, and lines 2-22 calculate the edge weight for all valid, non-base edges. Lines 3-5 initialize the weight and the remaining TEU ($\Delta\tau$) and remaining reefer capacity ($\Delta\tau^r$). The remaining capacities are initialised using the block capacities defined in Section 3.A. In line 6, the set $\mathcal{T}_{ij}$ is created. This set consists of the transports that can have cargo transported by the block if the considered edge is part of the longest path. The set $\mathcal{K}$ is a tuple with all the relevant transports and container types together with the value. We want to assign the most profitable containers first, and therefore the set $\mathcal{K}$ is sorted according to descending order of f^{tc} in line 8. In lines 9-21 elements from the set $\mathcal{K}$ are processed in order. Lines 10-14 calculate the capacity of the specific type depending on if it is a reefer or not. In line 15 we determine the actual number of containers of the considered type we can assign. Here we consider the demand (d^{tc}) as well as the capacity. We divide the capacity by the TEU coefficient (Γ^c), to have the actual number of

containers. Line 16 updates the edge weight and lines 17-20 update the remaining TEU capacity, $\Delta\tau$, and the remaining reefer capacity $\Delta\tau^r$. Lastly lines 23 and 24 respectively set the weight for the base and invalid edges.

The edge weights are set in such a way that the longest path in the graph corresponds to a block assignment where the structure of the demand is taken into account as well as the attributes of the specific blocks. The idea here is similar to what Christensen and Pacino (2017) proposed for the deterministic version of the cargo-mix problem. In Christensen and Pacino (2017) a full service is considered, and therefore the graph structure is different. Results showed that this outperformed a random method.

The longest path problem is solved for each block. The assigned cargo is then subtracted from the demand matrix to account for the cargo the generated schedule can carry. The edge weight calculations are based on the demand, and thus the edge weights are updated from block to block. Therefore, the longest path differs from block to block.

The graph G is a directed acyclic graph, and the longest path problem can be solved efficiently by topologically sorting the vertices and using a slightly modified version of Dijkstra's algorithm, such that the longest path is found instead of the shortest.

3.5.1.2 Including accepted bookings

The previously described procedure does not ensure that there is enough capacity assigned a transport t' to load the cargo that has been accepted for that transport. Assume some cargo has been accepted for the transport t' (i.e. $\sum_{c \in \mathcal{C}} u^{t'c} > 0$), then there must exists at least one block where port $d^{t'}$ is assigned as the discharge port for port $o^{t'}$. If this is not the case, the accepted cargo cannot be loaded, making it impossible to keep the promise to the customer.

Aside from the capacity considerations, the stability of the vessel will also need to be taken into account when ensuring the accepted cargo can be loaded. With accepted cargo assigned to blocks, it needs to be possible to load the rest of the ship in a way that makes the vessel stable, and the overall problem feasible.

Thus, before making the full block assignment (using the graph-based method), we want to partially fix the block assignment ensuring the accepted cargo can be transported. We want to do this alongside minimising the capacity usage the partially fixed block assignment takes up. The partially fixed block assignment can be satisfied in the graph-based method, by extending the set of invalid edges ($E_I(b)$).

The problem is formulated as mixed integer programming problem. First let the variable $y_b^{tc} \in \mathbb{R}^+$ be the number of containers of type $c \in \mathcal{C}$ of transport $t \in \mathcal{T}$ to be stowed in block $b \in \mathcal{B}$. This is similar to the variable y_{bs}^{tc} (from Section 3.4) but without the scenario index. Similar, define w_{bp} as the weight stowed in block $b \in \mathcal{B}$ at port $p \in \mathcal{P}^{l \setminus u}$. The main decision variable is σ_{bp}^d defining

the block assignment

$$\sigma_{bp}^{d} = \begin{cases} 1 & \text{If block } b \in \mathcal{B} \text{ is assigned destination } d \in \mathcal{P} \text{ at port } p \in \mathcal{P}^{l \setminus u} \\ 0 & \text{Otherwise.} \end{cases}$$

Along with the sets and parameters defined in Section 3.4, define $\mathcal{T}(d)$ as the set of transports with d as destination, for which containers has been accepted.

$$\mathcal{T}(d) = \left\{ t \in \mathcal{T} | d^t = d, \sum_{c \in \mathcal{C}} u^{tc} > 0 \right\}$$

and define $st(d)$ as the first port for which containers destined to d has been accepted

$$st(d) = \min_{t \in \mathcal{T}(d)} (o^t)$$

With this, the problem can be formulated as follows

$$\texttt{Min} \sum_{d \in \mathcal{P} \setminus p_0} \sum_{b \in \mathcal{B}} k_b^{TEU} \sigma_{bd-1}^{d} \tag{3.14}$$

`Subject to:`

$$\sum_{d \in \mathcal{P}} \sigma_{bp}^{d} \leq 1 \qquad \forall b \in \mathcal{B}, p \in \mathcal{P}^{l \setminus u} \tag{3.15}$$

$$\sigma_{bp}^{d^t} \geq \sigma_{bo^t}^{d^t} \qquad \forall b \in \mathcal{B}, t \in \mathcal{T}, \; o^t \leq p < \min(|\mathcal{P}^{l \setminus u}|, d^t) \tag{3.16}$$

$$\sigma_{bp}^{d} \geq \Omega_{bp}^{d} \qquad \forall b \in \mathcal{B}, p \in \mathcal{P}^{l \setminus u}, d \in \mathcal{P} \tag{3.17}$$

$$\mathbf{w} \in \mathcal{W} \tag{3.18}$$

$$\sum_{t \in \mathcal{T}_p^{ON}} \sum_{c \in \mathcal{C}} v^c y_b^{tc} \leq w_{bp} - \sum_{c \in \mathcal{C}} v^c \theta_b^{tc} \qquad \forall b \in \mathcal{B}, p \in \mathcal{P}^{l \setminus u} \tag{3.19}$$

$$\sum_{b \in \mathcal{B}} y_b^{tc} \leq \max \left(\min_{n \in \mathcal{N}} (a_n^{tc}), u^{tc} \right) \qquad \forall t \in \mathcal{T}, c \in \mathcal{C} \tag{3.20}$$

$$\sum_{b \in \mathcal{B}} y_b^{tc} \geq u^{tc} \qquad \forall t \in \mathcal{T}, c \in \mathcal{C} \tag{3.21}$$

$$\sum_{t \in \mathcal{T}(d)} \sum_{c \in \mathcal{C}} v^c y_b^{tc} \leq \left(q_b - \sum_{c \in \mathcal{C}} v^c \theta_{bst(d)}^{c} \right) \sigma_{bst(d)}^{d} \qquad \forall d \in \mathcal{P}, b \in \mathcal{B} \tag{3.22}$$

$$\sum_{t \in \mathcal{T}(d)} \sum_{c \in \mathcal{C}^{20}} y_b^{tc} \leq \left(k_b^{20} - \sum_{c \in \mathcal{C}^{20}} \theta_{bst(d)}^{c} \right) \sigma_{bst(d)}^{d} \qquad \forall d \in \mathcal{P}, b \in \mathcal{B} \tag{3.23}$$

$$\sum_{t \in \mathcal{T}(d)} \sum_{c \in \mathcal{C}} \Gamma^c y_b^{tc} \leq \left(k_b^{TEU} - \sum_{c \in \mathcal{C}} \Gamma^c \theta_{bst(d)}^{c} \right) \sigma_{bst(d)}^{d} \qquad \forall d \in \mathcal{P}, b \in \mathcal{B} \tag{3.24}$$

$$\sum_{t \in \mathcal{T}(d)} \sum_{c \in \mathcal{R}^{20}} y_b^{tc} \leq \left(r_b^{Slot} - \sum_{c \in \mathcal{R}^{20}} \theta_{bst(d)}^{c} \right) \sigma_{bst(d)}^{d} \qquad \forall d \in \mathcal{P}, b \in \mathcal{B} \tag{3.25}$$

$$\sum_{t\in\mathcal{T}(d)}\sum_{c\in\mathcal{R}}\tfrac{1}{2}\Gamma^c y_b^{tc} \leq \left(r_b^{Cell} - \sum_{c\in\mathcal{C}}\tfrac{1}{2}\Gamma^c\theta^c_{bst(d)}\right)\sigma^d_{bst(d)} \qquad \forall d\in\mathcal{P}, b\in\mathcal{B} \tag{3.26}$$

$$\sum_{t\in\mathcal{T}(d)}\sum_{c\in\mathcal{C}}\Phi^c y_b^{tc} \leq \left(h_b - \sum_{c\in\mathcal{C}}\Phi^c\theta^c_{bst(d)}\right)\sigma^d_{bst(d)} \qquad \forall d\in\mathcal{P}, b\in\mathcal{B} \tag{3.27}$$

$$0 \leq w_{bp} \leq q_b \qquad \forall b\in\mathcal{B}, p\in\mathcal{P}^{l\setminus u} \tag{3.28}$$

$$y_{bs}^{tc} \in \mathbb{R}^+ \qquad \forall b\in\mathcal{B}, t\in\mathcal{T}, c\in\mathcal{C} \tag{3.29}$$

$$\sigma^d_{bp} \in \{0,1\} \qquad \forall b\in\mathcal{B}, p\in\mathcal{P}^{l\setminus u}, d\in\mathcal{P} \tag{3.30}$$

The objective function (3.14) minimises the TEU capacity occupied by the partial block assignment. Here we consider σ^d_{bd-1} instead of σ^d_{bp} to only account for the capacity once for a destination. Constraints (3.15) - (3.17) define what should be satisfied for the block assignment, i.e at most one destination can be assigned a port (3.15), if d is chosen as the destination at port o, then all ports between o and d must have d as discharge port (3.16), and lastly constraint (3.17) ensures that the block assignment follows that imposed by the initial configuration. Constraints (3.18)-(3.27) ensure that the partially fixed block assignment results in an overall feasible problem. Constraints (3.18)-(3.19) ensure the stability of the vessel similar to how it is described in Section 3.4. In (3.20) we consider the number of containers available in the most conservative scenario, ensuring that the vessel will be stable in every scenario. Constraint (3.21) ensures that all the accepted cargo is assigned to a block on the vessel. Constraints (3.22)-(3.27) are capacity constraints, ensuring enough blocks are allocated to the transports for which we have accepted containers. These are reformulated capacity constraints, respectively the weight capacity, 20-foot capacity, TEU capacity, reefer slot capacity, reefer cell capacity and volume capacity (see Section 3.A for a description of the standard capacity constraints). The left-hand side of these constraints is the capacity usage of containers with a specific destination loaded in the considered block, and the right-hand side is the capacity. Here the capacity is the original capacity minus the capacity used by the initially loaded containers in that block. For a specific block and container type, the parameter θ^c_{bp} is non-increasing as p increases. Therefore we consider the port $st(d)$ when calculating the capacity used by the initially loaded containers, as this ensures there is enough capacity during the full journey.

In the above model, only containers from transports with accepted cargo must obey the block stowage requirement. Additionally, containers from transports with no accepted cargo are not accounted for, in the capacity constraints. This is intentional and helps to simplify the model. The main decision variables in the model are the block stowage variables, and the capacity constraints should only enforce we have enough capacity for the accepted cargo. We assume that if this model is feasible, then we can achieve a similar weight distribution from loading cargo satisfying the block stowage constraint and capacity constraints.

From the solution, we extract the partial block assignment which is used to make edges invalid in the graph-based method. Let $\Lambda(b)$ describe the partial block assignment for the block b.

$$\Lambda(b) = \{\langle p,d\rangle \in \mathcal{P}^{i\setminus u}\times\mathcal{P} | d\in\mathcal{P}, t\in\mathcal{T}(d), p = o^t, \sigma^d_{bp} = 1\} \qquad \forall b\in\mathcal{B}$$

Here $\langle p,d\rangle$ describes an assignment of d as discharge port for port p. From this we add the following set of edges to the set of invalid edges

$$\{(i,j)\in E | i < p, p > j, j \neq d\} \qquad \forall b\in\mathcal{B}, \langle p,d\rangle\in\Lambda(b) \tag{3.31}$$

$$\{(i,j) \in E | i = p, j \neq d\} \qquad \forall b \in \mathcal{B}, \langle p, d \rangle \in \Lambda(b) \qquad (3.32)$$

Consider a block b, and a block assignment $\langle p, d \rangle$, eq. (3.31) makes the edges originating at a port before p and terminating at a port after p invalid, except the edges terminating at port d. This ensures that either d is chosen as discharge port for port p, or the block is emptied at port p, making sure we can assign a new discharge port. If the block is emptied at port p, eq. (3.32) ensures port d is chosen as the discharge port, as it makes all other edges originating at p invalid.

3.5.2 Phase II

In Phase II the block assignment is fixed. Let $\widehat{\sigma}^d_{bp}$ be the block assignment found in Phase I, this is treated as an input parameter in the model, and will replace the σ^d_{bn} variables. Note that $\widehat{\sigma}^d_{bp}$ does not depend on the node, but only on the port, as discussed in Section 3.5.1. Phase II aims at finding a high-quality, feasible solution to the following problem.

$$\texttt{Max} \ (3.1) \qquad (3.33)$$

`Subject to:`

$$(3.2) - (3.5)$$
$$(3.9) - (3.12)$$
$$y^{tc}_{bs} \leq \max(a^{dc}_n, u^{tc}) \widehat{\sigma}^d_{bp^n} \qquad \forall n \in \mathcal{N}, s \in \mathcal{S}(n), b \in \mathcal{B}, c \in \mathcal{C}, d \in \mathcal{P}, t = \langle p^n, d \rangle \qquad (3.34)$$

Where constraint (3.34) ensures that the block assignment found in Phase I is satisfied and it is thus equivalent to constraint (3.8). Here we only need to check if the container can be loaded in its origin port. If this is the case, due to the construction of the block assignment, we know the block assignment will be ensured until its destination. This is not the case with constraint (3.8), as the block assignment is part of the decision.

By solving this model as is, we will still have the problem that the number of scenarios grows exponentially. The problem with a large number of scenarios is two-fold; first, the computational effort needed to solve the model will increase with the number of scenarios, second, memory problems might occur making it intractable to even build the model. To alleviate this, we will use a Rolling Horizon Heuristic (RHH).

The RHH solves the problem by decomposing the full problem into subproblems with shorter planning horizons. Doing so, the uncertainty in demand is only considered for the next few ports ahead. The solution to the subproblems is used to fix part of the solution for the full problem. There are multiple benefits in only considering a shorter planning horizon, e.g. the problems to be solved become smaller and can thus be solved more efficiently, and the subproblems can be solved in parallel, and thus take advantage of the power of modern computers. The main downside is that we will have more problems to solve.

A subproblem considers a subtree (T') of the full scenario tree and fixes the solution for the nodes in the subtree. Let $root(T') \in \mathcal{N}$ be the root of a subtree and let ρ be a parameter

describing the number of forthcoming ports to consider the stochasticity for, i.e. the horizon. For a subproblem, SCMPBS(T') the subtree T' contains the following set of nodes.

$$\mathcal{N}(T') = \left\{n \in \mathcal{N} | stage(root(T')) \leq stage(n) \leq stage(root(T')) + \rho, root(T') \in path(n)\right\}$$

Where $stage(n)$ is the stage associated with node $n \in \mathcal{N}$. Thus in the subproblem SCMPBS(T') we consider the nodes in the next ρ stages that has $root(T')$ on their path. Let $\mathcal{S}(T')$ be the scenarios considered in the problem SCMPBS(T'). Along with the nodes $\mathcal{N}(T')$, *artificial nodes* are considered as well. The artificial nodes make sure that solution does not only optimise over the set of ports considered in T', but that the forthcoming ports also are taken into account. For the artificial nodes, the uncertainty in demand is not considered, and the average is used as an estimate for the demand. For all forthcoming ports not considered in T' (all ports $p > p^{root(T')}+\rho$) a single artificial node is added for every scenario $s \in \mathcal{S}(T')$. After solving the subproblem SCMPBS(T') the solution is fixed for all nodes $n \in \mathcal{N}(T')$, and we continue with the next subproblem.

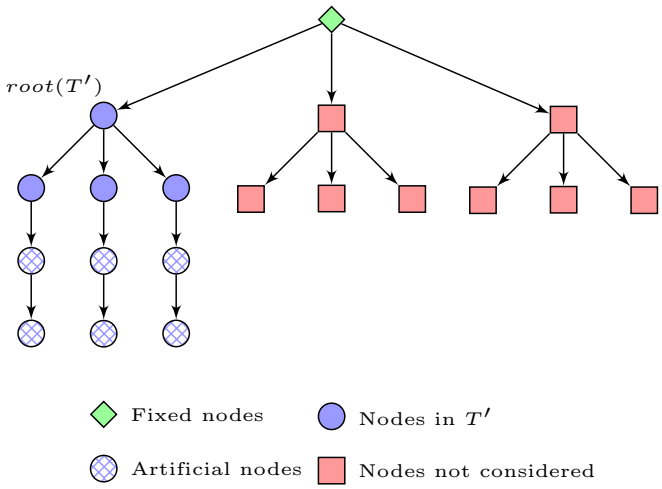
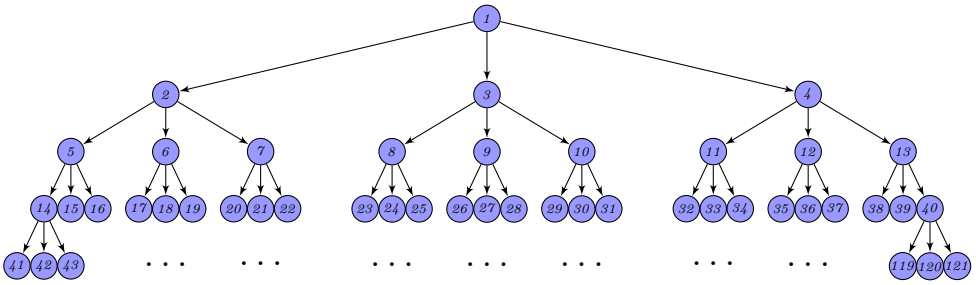


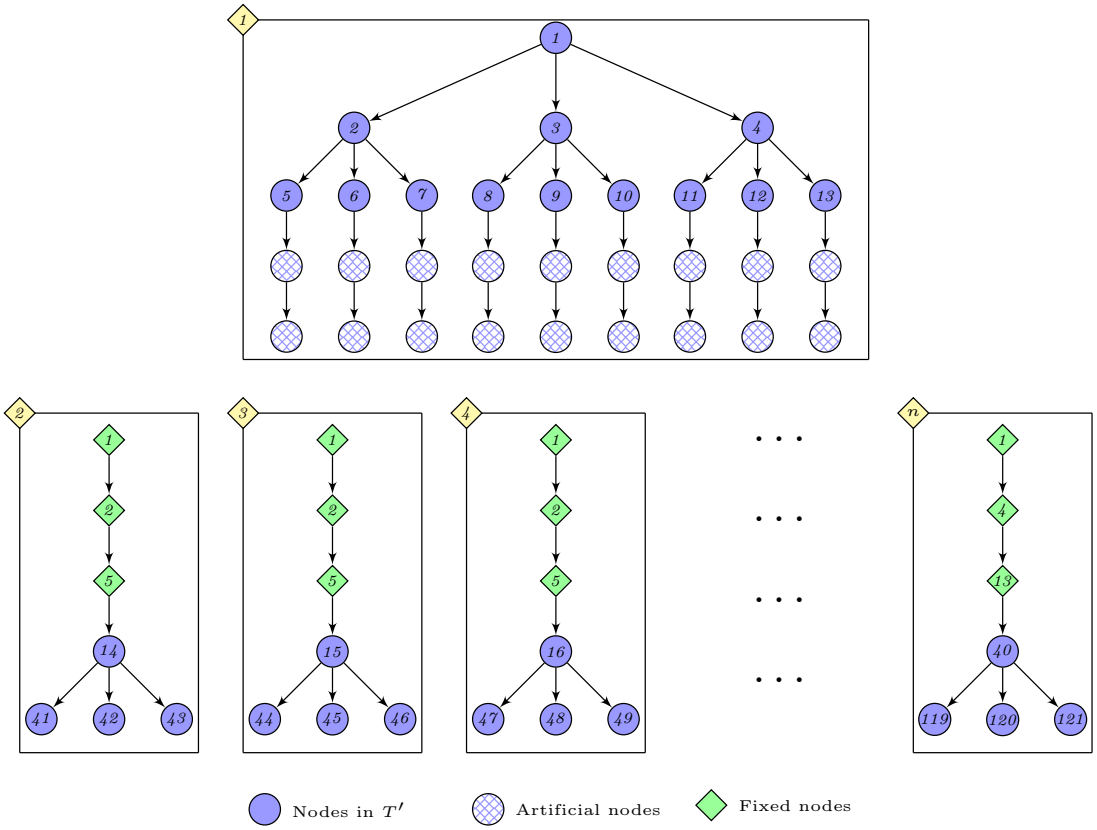
Figure 3.6: An example subtree with $\rho = 1$.

Figure 3.6 shows an example of a subtree T' with $\rho = 1$. In Figure 3.6 the solution for the diamond-shaped node has already been fixed by another subproblem. The solid circle nodes make up the set $\mathcal{N}(T')$, the striped nodes are the artificial nodes, and the square nodes are considered in another subproblem and are thus disregarded in the current.

The parameter ρ describes the subproblems we need to solve to get a full solution. For all the subtrees T' having a root node with the stage within the set $\{1, 1+(\rho+1), 1+2(\rho+1), \ldots, 1+|\mathcal{P}^{l\backslash u}|(\rho+1)\}$ a subproblem SCMPBS(T') will be solved. As example, if $\rho = 2$ and $|\mathcal{P}^{l\backslash u}| = 7$ we need to solve a subproblem for all nodes in the following stages: 1, 4, and 7. For the rest of the nodes, the solution is fixed in a previously solved subproblem. The subproblems are solved in order, starting with the subproblem with the lowest stage. Figure 3.7 shows an example of the RHH with $\rho = 2$. Figure 3.7a shows the scenario tree for the full problem, and Figure 3.7b shows how this is decomposed into a set of subproblems. In Figure 3.7b the solid circle nodes are the nodes $\mathcal{N}(T')$, the striped nodes are the artificial nodes, and the diamond-shaped nodes are the nodes for which the solution already has been fixed. The numbers within the nodes are used to identify the nodes from each other. The number in the top left corner is the subproblem number and shows the order in which the subproblems are solved.



(a) The full scenario tree for a problem with 5 ports to consider ($|\mathcal{P}^{l\setminus u}| = 5$).



(b) An example of the Rolling Horizon Heuristic with $\rho = 2$

Figure 3.7: Explanation of the Rolling Horizon Heuristic

The problem solved for subproblem SCMPBS(T') is the same as the problem (3.33)-(3.34), but on a redefined set of scenarios, nodes and ports. The parameter θ^c_{bp} is updated for every subproblem, to account for the part of the problem for which the solution has already been fixed.

The parameter ρ describes a tradeoff between the complexity of the subproblems, and the number of subproblems to solve. The higher a value of ρ, the more complex the subproblems are, a lower value means simpler subproblems, but more will have to be solved. However, a higher value of ρ should also imply a better solution.

With the rolling horizon heuristic, we can effectively handle the exponential growth in the number of scenarios. Instead, the number of subproblems to solve will grow exponentially, but we are

very unlikely to experience memory problems, and the RHH allows to solve the subproblems in parallel as the subproblems for a specific stage are independent of each other.

3.6 Data

Most liner shipping routes reflect the nature of an international trade, e.g. the Europe to Far East services. Here containers are exported from the Far East and imported to Europe. Vessels seldom docks at ports between the Suez canal, and the Singapore Strait. Due to the length of this leg, it is the most important concerning revenue optimisation. To reflect this, we consider 5 Europe-Far East/Far East-Europe services operated by our industry collaborator. These services will be the basis of our data. All of the services are being considered from both directions. For the Europe-Far East direction the ports in Europe are considered to be unloading/loading ports, where as only unload operations are planned for the ports in the Far East, and vice versa for the Far East-Europe direction. With this separation, we want to maximise the revenue on the most significant leg, while still considering the in-region revenue. To test how the solution method scales with respect to the number of loading ports, several instances are made out of each of the services. We control this by selecting different ports to what we will consider being the start. We restrict the number of loading ports to be considered to be greater than or equal to 3. The maximum number of loading ports to consider is determined by the services and is 9. In total 82 instances has been generated.

To generate the initial configuration of the vessel, a deterministic version of the problem is considered. Using the method described in Christensen and Pacino (2017) a solution to the deterministic problem is found. From the solution to the deterministic problem, 50% of the containers loaded on the vessel when it docks at the first loading port is removed, and with this imposing an initial discharge port for some blocks.

The base port-to-port demand and port drafts are based on data from LinerLib (Brouer et al., 2014), and multiplied with a scalar to be more suitable for the SCMPBS. Successively the total demand of each leg is decomposed into a demand for each of the specific container types by using the same predefined probabilities as in Christensen and Pacino (2017). The revenue of a dry 40-foot container is based on data from LinerLib, and from this, we calculate the revenue of each container type similar as to described in Delgado (2013). In a scenario, all the demand is multiplied with a scenario dependent scalar, such that the demand for all containers is 'high' in a 'high' scenario.

Table 3.1 describe how the scenario trees are generated. The scenarios for a port depends on the distance (in number of ports) from the start port. The farther away, the bigger the uncertainties are. The second column is the number of scenarios generated for the port. The third and fourth column describes the demand scenarios with the demand factor and the probabilities. The numbers in the third column describe the difference in percentage with respect to the expected demand.

Table 3.1: Scenario Tree generation description

Port distance	Number of scenarios	Demand scenarios	Probabilities
1	3	0, 2.5%, -2.5%	70%, 15%, 15%
2	3	0, 4%, -4%	60%, 20%, 20%
3	3	0, 5%, -5%	50%, 25%, 25%
4	3	0, 7.5%, -7.5%	40%, 30%, 30%
5	4	2.5%, -2.5%, 5%, -5%	35%, 35%, 15%, 15%
6	4	4%, -4%, 7.5, -7.5%	30%, 30%, 20%, 20%
7	5	0, 2.5%, -2.5%, 7.5%, -7.5%	10%, 30%, 30%, 15%, 15%
8	5	0, 5%, -5%, 10%, -10%	10%, 25%, 25%, 20%, 20%
9	5	0, 5%, -5%, 12.5%, -12.5%	10%, 22.5%, 22.5%, 22.5%, 22.5%

3.7 Computational Results

Three different solutions methods have been tested and compared.

- **Compact - SCMPBS Mixed Integer Programming model**
  The mixed integer programming model as presented in Section 3.4.
- **RHH - SCMPBS Rolling Horizon Heuristic**
  The rolling horizon heuristic as described in Section 3.5. All the subproblems are solved sequentially and not in parallel.
- **ModelHeu- SCMPBS Mathematical modelling based heuristic**
  This is a version of the heuristic where the model (3.33)-(3.34) is used instead of the rolling horizon heuristic. Phase I remains as described in Section 3.5.1.

All methods have been implemented in JAVA 1.8 and are tested using a 2.30 GHz Intel Xeon E5 processor and 128 GB memory. CPLEX v. 12.7.0 is used as the mixed-integer-programming solver. For the Compact and ModelHeu method, a time limit of 5 hours has been used. For all methods, an optimality tolerance of 0.05% was used.

An upper bound is calculated by relaxing the non-anticipativity constraints (3.4) in the compact model. Doing so there are no constraints linking decisions in one scenario to decisions in another scenario. Thus the problem can be decomposed by scenario. Each scenario-decomposed problem corresponds to a deterministic problem and can be solved independently from the rest.

Table 3.2 shows the overall results and compares the compact model with the two heuristics. For the RHH the horizon parameter ρ is 1. The two first columns show the number of ports and the number of instances in the instance class. For the three methods, #Sol is the number of instances where a feasible solution is found, the #Opt columns show the number of instances for which the model terminates with the optimal solution (For the ModelHeu method a solution is optimal if it is proved to be optimal for the fixed block assignment). The $\bar{x}$ columns are the average revenue in millions of dollars, *Gap* is the average relative difference between the solution and the calculated upper bound. Lastly $\bar{t}$ is the average execution time in seconds. In the table † means that the model terminated due to insufficient memory.

Table 3.2: Overall Results. Here † means that the model terminated due to insufficient memory.

		UB	Compact					RHH, $\rho = 1$				ModelHeu				
$\lvert P^{l\setminus u}\rvert$	n	$\bar{x}(10^6\$)$	#Sol	#Opt	$\bar{x}(10^6\$)$	*Gap*	$\bar{t}$	#Sol	$\bar{x}(10^6\$)$	*Gap*	$\bar{t}$	#Sol	#Opt	$\bar{x}(10^6\$)$	*Gap*	$\bar{t}$
3	18	15.06	18	9	13.74	11.88%	10451.6	18	14.83	1.73%	6.7	18	18	14.83	1.72%	91.1
4	17	17.89	17	3	9.85	45.38%	15888.5	17	17.64	1.50%	20.6	17	16	17.64	1.50%	4286.8
5	17	20.92	15	0	4.24	74.35%	18000.0	17	20.61	1.54%	58.4	17	4	19.99	7.34%	15375.6
6	12	23.42	0	0	†	†	†	12	23.10	1.61%	270.0	3	0	22.99	0.81%	18000.0
7	9	24.07	0	0	†	†	†	9	23.75	1.49%	463.5	0	0	†	†	†
8	6	23.76	0	0	†	†	†	6	23.41	1.77%	3278.8	0	0	†	†	†
9	3	20.03	0	0	†	†	†	3	19.64	2.46%	9239.1	0	0	†	†	†
Average		**19.89**			**9.56**	**42.01%**	**14569.1**		**19.60**	**1.63%**	**686.2**			**17.74**	**3.34%**	**7089.7**

With respect to the compact model, the results show what we would have expected; only the smallest of the instances can be solved to optimality but uses considerably more time than the two other methods. The quality of the solutions from the compact model quickly drops when increasing the instance size, and memory problem occurs for the medium to large sized instances. Overall, the compact method finds the optimal solution for 12 of the instances. For these instances, the average relative difference to the calculated upper bound is 0.15%. This suggests that the upper bound method finds upper bounds of high quality.

As it also can be seen from Table 3.2 the rolling horizon heuristic finds high-quality solutions within a reasonable amount of time. The solution quality is stable around 1.60% with only a few deviations.

For the 2-phased model based heuristic (ModelHeu), we see a similar behaviour as for the compact model; the smallest of the instances can be solved, but the largest ones become too big. Looking at the smallest of the instances (3 and four ports) the solution quality is comparable with the rolling horizon heuristic, but the RHH is much faster.

Table 3.3 analyses the impact of the horizon parameter ρ in the rolling horizon heuristic. Here $\rho = 0$ means that no stochasticity in the demand is considered in the subproblems. Each subproblem corresponds to a deterministic problem, which is solved to fix the solution in the root node. The results show that the quality of the solutions is not impacted by a change in the parameter ρ. However, the execution time is increased by incrementing ρ. Thus, the increase in the complexity of the subproblems, outweighs the decrease in the number of subproblems that needs to be solved.

Table 3.3: Rolling horizon heuristic results

		UB	RHH, $\rho = 0$			RHH, $\rho = 1$			RHH, $\rho = 2$		
$\lvert P^{l\setminus u}\rvert$	n	$\bar{x}(10^6\$)$	$\bar{x}(10^6\$)$	*Gap*	$\bar{t}$	$\bar{x}(10^6\$)$	*Gap*	$\bar{t}$	$\bar{x}(10^6\$)$	*Gap*	$\bar{t}$
3	18	15.06	14.83	1.73%	3.0	14.83	1.73%	6.7	14.83	1.72%	65.8
4	17	17.89	17.64	1.51%	7.3	17.64	1.50%	20.6	17.64	1.50%	121.9
5	17	20.92	20.61	1.54%	21.8	20.61	1.54%	58.4	20.61	1.54%	1027.1
6	12	23.42	23.09	1.61%	75.2	23.10	1.61%	270.0	23.10	1.60%	3858.1
7	9	24.07	23.75	1.49%	260.2	23.75	1.49%	463.5	23.75	1.48%	7595.6
8	6	23.76	23.40	1.78%	1218.3	23.41	1.77%	3278.8	23.41	1.77%	10738.6
9	3	20.03	19.64	2.47%	9435.2	19.64	2.46%	9239.1	19.59	2.86%	31324.6
Average		**19.89**	**19.60**	**1.63%**	**480.6**	**19.60**	**1.63%**	**686.2**	**19.60**	**1.64%**	**3582.7**

In the rolling horizon heuristic the block assignment is fixed in Phase I, and in Phase II the containers are stowed respecting the decisions from Phase I. Due to the splitting of the decisions,

both phases have an impact on the final solution quality. Table 3.4 analyses the two phases of the rolling horizon heuristic with respect to the solution quality. The table aims to explain how the final gap is split between the two phases. The UB columns is for the upper bound, and RHH is for the Rolling Horizon Heuristic. For the upper bound, $\bar{x}$ is an overall upper bound, and $\bar{x}_{II}$ is an upper bound given the fixed block assignment. The RHH result and $\bar{x}_{II}$ bound are based on the same block assignment, and thus $\bar{x}_{II}$ is a bound on the solution found in Phase II of the RHH. After fixing the block assignment, the $\bar{x}_{II}$ bound is calculated by decomposing the problem similarly as done when calculating the $\bar{x}$ bound as previously described. For the RHH, the first column is the value of the average solution. The next two columns are the gap to the two different upper bounds, and the last columns are the average time spent in the two phases.

Table 3.4: Rolling horizon heuristic Phase I and II Results

		UB		**RHH**, $\rho = 1$				
$\lvert P^{l\setminus u}\rvert$	n	$\bar{x}(10^6\$)$	$\bar{x}_{II}(10^6\$)$	$\bar{x}(10^6\$)$	Gap	Gap_{II}	$\bar{t}_I$	$\bar{t}_{II}$
3	18	15.06	14.83	14.83	1.73%	0.01%	0.6	6.1
4	17	17.89	17.64	17.64	1.51%	0.02%	0.7	19.9
5	17	20.92	20.62	20.61	1.54%	0.04%	1.4	56.9
6	12	23.42	23.10	23.09	1.61%	0.04%	1.3	268.8
7	9	24.07	23.76	23.75	1.49%	0.05%	1.5	462.0
8	6	23.76	23.41	23.40	1.78%	0.05%	2.0	3276.8
9	3	20.03	19.67	19.64	2.47%	0.18%	3.4	9235.6
Average		**19.89**	**19.61**	**19.60**	**1.63%**	**0.04%**	**1.2**	**685.0**

As also shown in Table 3.2 and Table 3.3, Table 3.4 shows that the average gap for the RHH is 1.63%. When comparing with the upper bound for the problem with the fixed block assignment, the average gap is only 0.04%, meaning a major part of the 1.63% of the overall gap stems from the solution to Phase I. Looking at the average time spent for each of the phases, we can see that Phase II is the most computational expensive phase, and the model for the accepted bookings in Phase I can easily be solved.

3.8 Conclusion

In this paper, the Stochastic Cargo Mix Problem is studied. The problem aims to find the cargo composition needed for a vessel to maximise its revenue on a given service. A solver for this problem can be a valuable analysis tool for the industry and can be used to perform various kinds of what-if analysis. For this sort of analysis, a fast solver is needed.

The results show that the mathematical model can only solve the smallest of the considered instances. Instead, a rolling horizon based matheuristic is developed. The matheuristic is shown to find high-quality solutions and fulfilling the need for a quick response time. Moreover, the matheuristic is scalable and can solve industrial size instances. The matheuristic is based on mathematical modelling techniques, and it is thus simple to add additional constraints, or remove existing constraints, making it easy to perform various what-if analysis.

The presented solution method is a two-phased matheuristic, and in the computational study, we experimentally prove that any improvement of the results should come from improving Phase I. Thus an idea for further research is to improve the way the block assignment is computed in Phase

I e.g. by making the block assignment scenario dependent. Phase II is the most computationally expensive part of the method. However, the subproblems solved in Phase II can be solved in parallel, and the execution of the method can thus be sped up by utilising this.

Furthermore, we plan to study how this work can be combined with other essential problems in the liner shipping industry, for example, cargo flow optimisation, service network design or empty re-positioning.

References

Ambrosino, D., D. Anghinolfi, M. Paolucci, and A. Sciomachen (2010). "An Experimental Comparison of Different Heuristics for the Master Bay Plan Problem". In: *Experimental Algorithms*. Ed. by P. Festa. Vol. 6049. Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 314–325.

Ambrosino, D., M. Paolucci, and A. Sciomachen (2015a). "A MIP Heuristic for Multi Port Stowage Planning". In: *Transportation Research Procedia* 10. 18th Euro Working Group on Transportation, EWGT 2015, 14-16 July 2015, Delft, The Netherlands, pp. 725–734. ISSN: 2352-1465.

Ambrosino, D., M. Paolucci, and A. Sciomachen (2015b). "Computational evaluation of a MIP model for multi-port stowage planning problems". In: *Soft Computing*, pp. 1–11. ISSN: 1433-7479.

Ambrosino, D., A. Sciomachen, and E. Tanfani (2004). "Stowing a containership: the master bay plan problem". In: *Transportation Research Part A: Policy and Practice* 38.2, pp. 81–99. ISSN: 09658564.

Botter, R. C. and M. A. Brinati (1992). "Stowage Container Planning: A Model for Getting an Optimal Solution". In: *Proceedings of the IFIP TC5/WG5.6 Seventh International Conference on Computer Applications in the Automation of Shipyard Operation and Ship Design, VII*. Amsterdam, The Netherlands, The Netherlands: North-Holland Publishing Co., pp. 217–229. ISBN: 0-444-89728-3.

Bredstrom, D. and M. Rönnqvist (2006). "Supply Chain Optimization in Pulp Distribution Using a Rolling Horizon Solution Approach". In: *Norwegian School of Economics Dept. of Finance & Management Science Discussion Paper No. 2006/17*.

Brouer, B. D., J. F. Alvarez, C. E. M. Plum, D. Pisinger, and M. M. Sigurd (2014). "A Base Integer Programming Model and Benchmark Suite for Liner-Shipping Network Design". In: *Transportation Science* 48, pp. 281–312. ISSN: 0041-1655.

Chand, S., V. N. Hsu, and S. Sethi (2002). "Forecast, Solution, and Rolling Horizons in Operations Management Problems: A Classified Bibliography". In: *Manufacturing & Service Operations Management* 4.1, pp. 25–43.

Christensen, J. and D. Pacino (2017). "A matheuristic for the Cargo Mix Problem with Block Stowage". In: *Transportation Research Part E: Logistics and Transportation Review* 97, pp. 151–171. ISSN: 1366-5545.

Delgado, A. (2013). "Models and Algorithms for Container Vessel Stowage Optimization".

Feng, C.-M. and C.-H. Chang (2008). "Optimal Slot Allocation in Intra-Asia Service for Liner Shipping Companies". In: *Maritime Economics & Logistics* 10.3, pp. 295–309. ISSN: 1479-2931.

Kang, J.-G. and Y.-D. Kim (2002). "Stowage planning in maritime container transportation". In: *Journal of the Operational Research Society* 53.4, pp. 415–426.

Li, F., C. Tian, R. Cao, and W. Ding (2008). "An integer linear programming for container stowage problem". In: *Computational Science–ICCS 2008*, pp. 853–862.

Pacino, D., A. Delgado, R. M. Jensen, and T. Bebbington (2011). "Fast Generation of Near-Optimal Plans for Eco-Efficient Stowage of Large Container Vessels". In: *Computational Logistics*. Ed. by J. W. Böse, H. Hu, C. Jahn, X. Shi, R. Stahlbock, and S. Voß. Vol. 6971. Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 286–301.

Pacino, D., A. Delgado, R. Jensen, and T. Bebbington (2012). "An Accurate Model for Seaworthy Container Vessel Stowage Planning with Ballast Tanks". English. In: *Computational Logistics*. Ed. by H. Hu, X. Shi, R. Stahlbock, and S. Voß. Vol. 7555. Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 17–32. ISBN: 978-3-642-33586-0.

Rakke, J. G., M. Stålhane, C. R. Moe, M. Christiansen, H. Andersson, K. Fagerholt, and I. Norstad (2011). "A rolling horizon heuristic for creating a liquefied natural gas annual delivery program". In: *Transportation Research Part C: Emerging Technologies* 19.5, pp. 896–911.

Wilson, I. and P. Roach (2000). "Container Stowage Planning: A Methodology for Generating Computerised Solutions". In: *The Journal of the Operational Research Society* 51.11, pp. 1248–1255.

3.A Capacity Constraints

To model the capacity constraints, we must introduce additional sets. First let $\mathcal{R} \subset \mathcal{C}$ be the set of reefer containers, additional let $\mathcal{C}^{20} \subset \mathcal{C}$ be the set of containers with length 20 feet. The sets and additional parameters are summarised below.

Sets:

$\mathcal{C}^{20} \subset \mathcal{C}$	Set of container types with length 20 feet.
$\mathcal{R} \subset \mathcal{C}$	Set of reefer container types
$\mathcal{R}^{20} \subset \mathcal{R}$	Set of reefer container types, with length 20 feet
$\mathcal{T}_p^{ON}$	Set of transports that visits port $p \in \mathcal{P}$

Parameters:

$\theta_{bp}^c \in \mathbb{N}$	Number of containers of type $c \in \mathcal{C}$ initially loaded that still occypy block $b \in \mathcal{B}$ at port $p \in \mathcal{P}$
$k_b^{TEU} \in \mathbb{N}$	Teu capacity for block $b \in \mathcal{B}$
$k_b^{20} \in \mathbb{N}$	Capacity for 20' containers for block $b \in \mathcal{B}$
$r_b^{Cell} \in \mathbb{N}$	Reefer cell capacity of block $b \in \mathcal{B}$
$r_b^{Slot} \in \mathbb{N}$	Reefer slot capacity of block $b \in \mathcal{B}$
$\Gamma^c \in \{1, 2\}$	TEU coefficient of container type $c \in \mathcal{C}$
$h_b \in \mathbb{R}^+$	Volume capacity for block $b \in \mathcal{B}$
$\Phi^c \in \mathbb{R}^+$	Volume coefficient of container type $c \in \mathcal{C}$

$$\sum_{t \in \mathcal{T}_p^{ON}} \sum_{c \in \mathcal{C}^{20}} y_b^{tc} \leq k_b^{20} - \sum_{c \in \mathcal{C}^{20}} \theta_{bp}^c \qquad \forall b \in \mathcal{B}, p \in \mathcal{P}^{l \setminus u} \qquad (3.35)$$

$$\sum_{t \in \mathcal{T}_p^{ON}} \sum_{c \in \mathcal{C}} \Gamma^c y_b^{tc} \leq k_b^{TEU} - \sum_{c \in \mathcal{C}} \Gamma^c \theta_{bp}^c \qquad \forall b \in \mathcal{B}, p \in \mathcal{P}^{l \setminus u} \qquad (3.36)$$

$$\sum_{t \in \mathcal{T}_p^{ON}} \sum_{c \in \mathcal{R}^{20}} y_b^{tc} \leq r_b^{Slot} - \sum_{c \in \mathcal{R}^{20}} \theta_{bp}^c \qquad \forall b \in \mathcal{B}, p \in \mathcal{P}^{l \setminus u} \qquad (3.37)$$

$$\sum_{t \in \mathcal{T}_p^{ON}} \sum_{c \in \mathcal{R}} \tfrac{1}{2} \Gamma^c y_b^{tc} \leq r_b^{Cell} - \sum_{c \in \mathcal{R}} \tfrac{1}{2} \Gamma^c \theta_{bp}^c \qquad \forall b \in \mathcal{B}, p \in \mathcal{P}^{l \setminus u} \qquad (3.38)$$

$$\sum_{t \in \mathcal{T}_p^{ON}} \sum_{c \in \mathcal{C}} \Phi^c y_b^{tc} \leq h_b - \sum_{c \in \mathcal{C}} \Phi^c \theta_{bp}^c \qquad \forall b \in \mathcal{B}, p \in \mathcal{P}^{l \setminus u} \qquad (3.39)$$

Equations (3.35)-(3.39) are the block capacity constraints. For each block of the vessel, we distinguish between a TEU and a 20-foot capacity. This is due to the layout of the vessel since not all the blocks can stow 20-foot containers. In most of the cases k_b^{20} is, however, equal to k_b^{TEU}. Constraints (3.35) and (3.36) ensure that both of these capacities are satisfied. Similar to the TEU capacity constraints, the reefer constraints are split into a slot capacity constraint and a cell capacity constraint. Each reefer container needs to be plugged into an electrical output to get power for the cooling unit and r_b^{Slot} describes the number of such slots in a block b. A reefer cell is a cell where one of the slots has a reefer plug, and this number is denoted by r_b^{Cell}, if both slots in a reefer cell have a reefer plug, then $r_b^{Slot} = 2r_b^{Cell}$. In more general terms $r_b^{Cell} \leq r_b^{Slot} \leq 2r_b^{Cell}$ will be satisfied for all blocks. Constraint (3.37) restricts the number of reefer slots that can be used, and (3.38) is the reefer cell capacity constraint. Here we multiply with $\frac{1}{2}\Gamma^c$ as a 40-foot container occupies a full cell, whereas a 20-foot container occupies half a cell, and due to (3.37) the slot capacity will be satisfied. Constraint (3.39) limits the total volume of the containers in a block. This is needed to account for the fact that high-cube containers are higher and thus use more than a simple TEU capacity.

The set of constraints (3.35)-(3.39) defines a polyhedron with the feasible stowage plans. For the variables in the vector $\mathbf{y}_s$, this polyhedron is denoted by $\mathcal{Y}$ in the model in Section 3.4.

3.B Stability Constraints[1]

The stability constraints ensure that the vessel does not capsize or break, even in tough weather conditions. The constraints here uses similar notation as the one introduced in Delgado (2013), and the interested reader is invited to look at the source for a more elaborate description.

The main difference between the set of constraints presented below and (6.25) - (6.43) of Delgado (2013), is the draft. Delgado (2013) does not consider draft limit for each port, thus making it possible for the vessel to have a greater displacement. The introduction of the LinerLib (Brouer et al., 2014) makes this data easily available, and the draft limit is therefore included in this work. Also, Delgado considers locations, where this work considers blocks.

The functions describing the metacentre, trim, draft and buoyancy force all depends non-linearly on the displacement. However, Delgado (2013) shows that these functions can be approximated

[1]The stability constraints and explanation thereof in this section is identical to Appendix A of Christensen and Pacino (2017) and is only included here for the reader's convenience

by linear planes by splitting the full displacement range into *displacement intervals*. A displacement interval is thus defined as a minimum, maximum (W_i^- and W_i^+) and an average (W_i) weight for the interval.

The sets, variables and parameters used in the model are introduced and explained below.

Sets:

$\mathcal{P}$	Set of ports.
$\mathcal{T}$	Set of ballast tanks.
$\mathcal{B}$	Set of blocks.
$\mathcal{I}$	Set of displacement intervals.
$\mathcal{BS}$	Set of bonjean stations.
$\mathcal{F}$	Set of frames.

Decision Variables:

w_{bp}	$\in \mathbb{R}^+$	Weight stowed in block $b \in \mathcal{B}$ at port $p \in \mathcal{P}$.
x_{tp}	$\in \mathbb{R}^+$	Weight of the tank $t \in \mathcal{T}$ at port $p \in \mathcal{P}$.

Auxiliary Variables:

v_p^W	$\in \mathbb{R}^+$	Vessel Displacement at port $p \in \mathcal{P}$.
v_{ip}^W	$\in \mathbb{R}^+$	Vessel Displacement in interval $i \in \mathcal{I}$ at port $p \in \mathcal{P}$.
v_{ip}^L	$\in \mathbb{R}$	Longitudinal centre of gravity at displacement interval $i \in \mathcal{I}$ at port $p \in \mathcal{P}$.
ψ_{ip}	$\in \{0,1\}$	Indicator variable for displacement interval $i \in \mathcal{I}$ at port $p \in \mathcal{P}$.
v_p^V	$\in \mathbb{R}^+$	Vertical centre of gravity at port $p \in \mathcal{P}$.
v_p^{VM}	$\in \mathbb{R}^+$	Vertical moment at port $p \in \mathcal{P}$.
v_p^M	$\in \mathbb{R}^+$	Metacentre at port $p \in \mathcal{P}$.
v_{bsp}^{Bs}	$\in \mathbb{R}^+$	Buoyancy force of section between bonjean station bs and $bs+1$ at port p.
$v_{f\alpha p}^S$	$\in \mathbb{R}^+$	Shear force fore or aft of frame $f \in \mathcal{F}$ at port $p \in \mathcal{P}$.
$v_{f\alpha p}^B$	$\in \mathbb{R}^+$	Bending moment fore or aft of frame $f \in \mathcal{F}$ at port $p \in \mathcal{P}$.

Parameters:

Various parameters:

W^O	Weight of the empty vessel.
Min^{GM}	Lower bound for metacentric height
Max_p^D	Maximum draft allowed at port p

Displacement intervals parameters:

$W_i^{\{-,+\}}$	Lower (-) and upper (+) bound of displacement interval $i \in \mathcal{I}$
W_i	Average weight of displacement interval $i \in \mathcal{I}$
$A^W_{\{M,T,D,Bs\}}(W_i)$	Weight coefficient of displacement interval $i \in \mathcal{I}$ for the linearization of metacentre (M), trim (T), draft (D), and bonjean at station bs (Bs).
$A^L_{\{M,T,Bs\}}(W_i)$	Lcg coefficient of displacement interval $i \in \mathcal{I}$ for the linearization of metacentre (M), trim (T), and bonjean at station bs (Bs).
$A_{\{M,T,D,Bs\}}(W_i)$	Constant of displacement interval $i \in \mathcal{I}$ for the linearization of metacentre (M), trim (T), draft (D), and bonjean at station bs (Bs).

Centre of gravity parameters:

$\text{Min}_i^L/\text{Max}_i^L$	Min/maximum longitudinal centre of gravity at displacement interval $i \in \mathcal{I}$
$D_b^{\{L,V\}}$	Longitudinal (L), and Vertical (V) centre of gravity of block $b \in \mathcal{B}$
$D_t^{\{L,V,T\}}$	Longitudinal (L), Vertical (V), and transversal (T) centre of gravity of ballast tank $t \in \mathcal{T}$
Max^V	Maximum vertical moment possible for the vessel.
LM^O	Longitudinal moment of the empty vessel including constant weights.
VM^O	Vertical moment of the empty vessel including constant weights.
TM^O	Transversal moment of the empty vessel including constant weights.

Bending/Shearing parameters:

$W^S_{f\alpha}$	Constant weights fore or aft of frame $f \in \mathcal{F}$
$G^\alpha_{\{b,t,bs\}f}$	Fraction of block b, ballast tank t, and buoyancy section between bonjean stations bs and $bs+1$ that lies fore or aft frame f
$W^B_{f\alpha}$	Bending components of the constant weight fore or aft of frame $f \in \mathcal{F}$
D_b^{Bs}	Distance in meters between bonjean stations bs and $bs+1$ multiplied by the density of water
$A^\alpha_{\{b,\ t,\ bs\}f}$	Fore or aft distance from frame f to the longitudinal centre of gravity of block b, ballast tank t, buoyancy section between bonjean stations bs and $bs+1$
G_f	Fore-based fraction of frame $f \in \mathcal{F}$, where $G_f \in [0;1]$. $G_f = 1$ when f is is the first frame at the bow, and $G_f = 0$ when f is the first frame at the stern.
$\text{Min}_f^{\{S,B\}}$	Lower bound for shear force (S) and bending moment (B) at frame $f \in \mathcal{F}$
$\text{Max}_f^{\{S,B\}}$	Upper bound for shear force (S) and bending moment (B) at frame $f \in \mathcal{F}$

With these the stability constraints can be modelled as seen below.

$$\sum_{t \in \mathcal{T}} x_{tp} + \sum_{b \in \mathcal{B}} w_{bp} + W^O = v_p^W \qquad \forall p \in \mathcal{P} \tag{3.40}$$

$$\sum_{i \in \mathcal{I}} W_i^- \psi_{ip} \leq v_p^W \leq \sum_{i \in \mathcal{I}} W_i^+ \psi_{ip} \qquad \forall p \in \mathcal{P} \qquad (3.41)$$

$$\sum_{i \in \mathcal{I}} \psi_{ip} = 1 \qquad \forall p \in \mathcal{P} \qquad (3.42)$$

$$\sum_{i \in \mathcal{I}} v_{ip}^W = v_p^W \qquad \forall p \in \mathcal{P} \qquad (3.43)$$

$$W_i^- \psi_{ip} \leq v_{ip}^W \leq W_i^+ \psi_{ip} \qquad \forall i \in \mathcal{I}, p \in \mathcal{P} \qquad (3.44)$$

$$\text{Min}_i^L \psi_{ip} \leq v_{ip}^L \leq \text{Max}_i^L \psi_{ip} \qquad \forall i \in \mathcal{I}, p \in \mathcal{P} \qquad (3.45)$$

$$\sum_{b \in \mathcal{B}} D_b^L w_{bp} + \sum_{t \in \mathcal{T}} D_t^L x_{tp} + LM^O = \sum_{i \in \mathcal{I}} W_i v_{ip}^L \qquad \forall p \in \mathcal{P} \qquad (3.46)$$

$$v_p^V W_i + (1 - \psi_{ip}) \text{Max}^V \geq v_p^{VM} \qquad \forall i \in \mathcal{I}, p \in \mathcal{P} \qquad (3.47)$$

$$v_p^V W_i - (1 - \psi_{ip}) \text{Max}^V \leq v_p^{VM} \qquad \forall i \in \mathcal{I}, p \in \mathcal{P} \qquad (3.48)$$

$$\sum_{b \in \mathcal{B}} D_b^V w_{bp} + \sum_{t \in \mathcal{T}} D_t^V x_{tp} + VM^O = v_p^{VM} \qquad \forall p \in \mathcal{P} \qquad (3.49)$$

$$\sum_{t \in \mathcal{T}} D_t^T x_{tp} + TM^O = 0 \qquad \forall p \in \mathcal{P} \qquad (3.50)$$

$$\sum_{i \in \mathcal{I}} A_M^W(W_i) v_{ip}^W + A_M^L(W_i) v_{ip}^L + A_M(W_i) \psi_{ip} = v_p^M \qquad \forall p \in \mathcal{P} \qquad (3.51)$$

$$v_p^M - v_p^V \geq \text{Min}^{GM} \qquad \forall p \in \mathcal{P} \qquad (3.52)$$

$$\sum_{i \in \mathcal{I}} A_T^W(W_i) v_{ip}^W + A_T^L(W_i) v_{ip}^L + A_T(W_i) \psi_{ip} = 0 \qquad \forall p \in \mathcal{P} \qquad (3.53)$$

$$\sum_{i \in \mathcal{I}} A_D^W(W_i) v_{ip}^W + A_D(W_i) \psi_{ip} \leq \text{Max}_p^D \qquad \forall p \in \mathcal{P} \qquad (3.54)$$

$$\sum_{s \in \{bs, bs+1\}} \sum_{i \in \mathcal{I}} A_{Bs}^W(W_i) v_{ip}^W + A_{Bs}^L(W_i) v_{ip}^L + A_{Bs}(W_i) \psi_{ip} = 2 D_d^{Bs} v_{bsp}^{Bs} \qquad \forall bs \in \mathcal{BS}, p \in \mathcal{P} \qquad (3.55)$$

$$W_{f\alpha}^S + \sum_{b \in \mathcal{B}} G_{bf}^\alpha w_{bp} + \sum_{t \in \mathcal{T}} G_{tf}^\alpha x_{tp} - \sum_{bs \in \mathcal{BS}} G_{bsf}^\alpha v_{bsp}^{Bs} = v_{f\alpha p}^S \qquad \forall f \in \mathcal{F}, \alpha \in \{A, F\}, p \in \mathcal{P} \qquad (3.56)$$

$$W_{f\alpha}^B + \sum_{b \in \mathcal{B}} A_{bf}^\alpha G_{bf}^\alpha w_{bp} + \sum_{t \in \mathcal{T}} A_{tf}^\alpha G_{tf}^\alpha x_{tp} - \sum_{bs \in \mathcal{BS}} A_{bsf}^\alpha G_{bsf}^\alpha v_{bsp}^B = v_{f\alpha p}^B \qquad \forall f \in \mathcal{F}, \alpha \in \{A, F\}, p \in \mathcal{P} \qquad (3.57)$$

$$\text{Min}_f^S \leq G_f v_{f,Fore,p}^S + (1 - G_f) v_{f,Aft,p}^S \leq \text{Max}_f^S \qquad \forall f \in \mathcal{F}, p \in \mathcal{P} \qquad (3.58)$$

$$\text{Min}_f^B \leq G_f v_{f,Fore,p}^B + (1 - G_f) v_{f,Aft,p}^B \leq \text{Max}_f^B \qquad \forall f \in \mathcal{F}, p \in \mathcal{P} \qquad (3.59)$$

Constraint (3.40) calculates the displacement of the vessel for every port. The next constraint, (3.41), sets the displacement interval variables together with (3.42) which ensures that exactly one displacement interval is active at each port. Constraint (3.43) and (3.44) defines v_{ip}^W to be equal to the displacement for the active displacement interval, and 0 for the rest. In a similar fashion (3.45) and (3.46) calculates the longitudinal centre of gravity (LCG). The centre of gravity is calculated as the sum of moments divided by the total displacement. The left-hand

side of (3.46) calculates the sum of moments. This is done by considering the longitudinal centre of gravity for the tanks and blocks and multiplying with the weight stowed in these. The right-hand side uses the average weight of the displacement interval, instead of the actual displacement. Similar with v_{ip}^W v_{ip}^L is zero for the displacement intervals for which $\psi_i = 0$ and for the active displacement interval it lies within the bounds defined by constraint (3.45). Constraint (3.47)-(3.49) approximates the vertical centre of gravity (VCG). Constraint (3.47) and (3.48) defines bound for the vertical moment for each displacement interval, and in the case when $\psi_i = 1$ the two inequalities turn into an equality $v_p^V W_i = v_p^{VM}$. For the non-active displacement intervals, these two constraints have no effect. Constraint (3.49) calculates the vertical moment similar to how the LCG is calculated, but without multiplying the right-hand side with the weight. (3.50) ensures that the transversal centre of gravity is 0, meaning the middle of the vessel. Due to the construction of the blocks, the transversal centre of gravity is 0 for all blocks, and thus only the tanks and the moment of the empty vessel are considered in the calculation. Constraint (3.51)-(3.55) calculates the metacentre, trim, draft and buoyancy force using the linearization of the non-linear functions. Each of the planes for the functions is described using three factors, $A^W(W_i)$, $A^L(W_i)$ and $A(W_i)$. $A^W(W_i)$ is the displacement factor, $A^L(W_i)$ is the LCG factor, and $A(W_i)$ is the constant factor. (3.51) calculates the metacentre, and (3.52) defines the metacentric height to be greater than the minimum metacentric height allowed. In (3.53) the trim is required to be zero, and (3.54) enforces the draft be less than or equal to Max_p^D. As the trim is required to be zero, the draft does not depend on the LCG, but only the displacement of the vessel. Max_p^D is the minimum draft allowed when leaving port p, and will thus be the minimum of the draft at port p and port $p+1$. Constraint (3.55) calculates the buoyancy force (*bonjean*) between station bs and $bs+1$. The last four constraints (3.56)-(3.59) are related to the stress forces. The first two calculates the shearing and bending, and (3.58)-(3.59) defines the upper and lower bounds. The shear force on a vessel, at a given frame, is the integral of forces on either side of the frame, and the bending moment is the integral of moments on either side of the frame. The buoyancy forces are only approximated, and thus there is an accumulation of error when calculating the shear force and the bending moment. To reduce the impact of this error, constraint (3.56) and (3.57) respectively calculates the shear forces and bending moment with respect to the resulting forces acting fore and aft of the frame. Hence there are two shear variables for every frame at each port. Constraint (3.58) and (3.59) respectively sets the limits for the shear force and bending moment at each frame. The shear force and bending moment at a frame are estimated as a proportional calculation based on the position of the frame. This reduces the impact of the error accumulation as the fore-based computation is accurate in the bow and the aft-based computation is accurate in the stern. All these constraints ensure that the vessel is stable and can be declared seaworthy if the stacking rules are obeyed.

The set of constraints (3.40)-(3.59) defines a polyhedron with the feasible weight allocations. For the variables in the vector $\mathbf{w}_s$, This polyhedron is denoted by $\mathcal{W}$ in the model in Section 3.4.

Chapter 4

Flexible ship loading problem with transfer vehicle assignment and scheduling

Çağatay Iris[a,b] · Jonas M. Christensen[a] · Dario Pacino[a] · Stefan Ropke[a]

[a]Management Science, Department of Management Engineering, Technical University of Denmark, Produktionstorvet, Building 424, DK-2800 Kgs. Lyngby, Denmark

[b]School of Civil and Environmental Engineering, Nanyang Technological University, 639798, Singapore, Singapore

Publication Status: Submitted to Transportation Research: Part B

Abstract: This paper presents the flexible containership loading problem for seaport container terminals. The integrated management of loading operations, planning of the equipment to use and their scheduling is what we define as the Flexible Ship Loading Problem (FSLP). The flexibility comes from a cooperative agreement between the terminal operator and the liner shipping company, specifying that the terminal has the right to decide which specific container to load for each slot obeying the class-based stowage plan received from the liner. We formulate a mathematical model for the problem. Then we present various modelling enhancements and a mathematical model to obtain strong lower bounds. We also propose a heuristic algorithm to solve the problem. It is shown that enhancements improve the performance of formulation significantly, and the heuristic efficiently generates high-quality solutions. Results also point out that substantial cost savings can be achieved by integrating the ship loading operations.

Keywords: Maritime logistic · Terminal operations · Ship Loading problem · GRASP

4.1 Introduction

Maritime freight transport constitutes an important part of the global logistics systems. Benefiting from rapid globalization, the containerized freight transport has been steadily growing over the past decade apart from the year 2009 with global financial crisis. The leading 100 container terminals have handled 539.2 million Twenty Equivalent Units (TEUs) in 2015 (UNCTAD (2015)) with an increase by 6.8% from 2014. Therefore, the increasing container handling volumes make operations planning a more complex and significant challenge for container terminals.

Liner shipping companies have adapted to the growth in the transport volumes by increasing the capacity of their services. This is done by deploying larger vessels of over 20.000 TEUs and planning more frequent visits to the containers terminals. Capacity is, however, not enough. A reliable shipping service requires the cargoes to arrive on time, so container terminals are required to supply reliable and agile operations for their customers. The increase in vessels size intensifies the pressure on the container terminals. Meanwhile, shipping companies also expect terminals to minimize the vessel turnaround (handling) times.

Vessel turnaround times might be reduced by deploying more Quay Cranes (QCs) and Transfer Vehicles (TVs) on each vessel, however, this does not guarantee an improvement in the service quality. There is a limited number of equipment that can be assigned to a vessel. Also, inefficient management of this equipment can bring more congestion and deterioration in the overall performance. Considering that QCs and TVs are limited resources with high operating costs, terminals should rather optimise the use of these resources.

We refer readers to the literature reviews on decision problems in seaside operations (Carlo et al. (2013), Bierwirth and Meisel (2015)), transport operations (Steenken et al. (2004), Carlo et al. (2014b)) and yard operations (Li and Vairaktarakis (2004), Carlo et al. (2014a)) in terminals. Literature reviews such as Kim and Lee (2015) note that there is a need for flexibility in operations, and possible collaboration with the liner shipping company can bring some flexibility in the ship loading related operations.

The efficient loading of containers to the vessel has become a more complicated problem due to the increase in vessel size, vessel numbers and complex technicalities. The high degree of industrial requirements (e.g. lashing patterns, vessel stress forces and staff working hour regulations) along with all other mentioned challenges, make efficient ship loading an even more complicated problem. It also often happens that some of the containers are ready to be loaded earlier but have to wait since they would be out of the planned load sequence. Due to the mentioned complexities and limited handling equipment, most attempts at improving the loading operations should be based on optimisation methods.

Some liner shipping companies are aware of the challenges that container terminals face, and have actively started to adapt their *stowage plans* to be more terminal friendly. A stowage plan describes the arrangement of containers on the vessel. In recent years, there has been a shift in the stowage planning policy which is based on an increasing collaboration between the terminal and the liner shipper. The liner provides the terminal with the stowage plan based on container classes (a container class is defined by the port of discharge, physical container dimensions, weight, etc.) which we refer to as *class-based stowage plan*. The terminal has the flexibility of

determining the position of specific containers of the same class obeying the class-based stowage plan (Monaco et al. (2014)). In this study, we integrate the assignment and scheduling of transfer vehicles and container load sequencing with the assignment of specific containers to the vessel positions. We call the entire problem the FSLP. We aim at reducing service times of the handling equipment and meeting the deadlines on the finishing time of the loading.

The contribution of the study is multi-fold. First, we introduce a new integrated container terminal problem to improve the efficiency of the loading operations. We formulate a mathematical model to solve the problem and some enhancements to improve this formulation. Then we suggest a model to obtain lower bounds for the problem. We also propose a heuristic method to solve it. Computational results show that the enhancements on the model significantly improve its performance, but still, the mathematical model is intractable for large scale instances. The results for the heuristic show that it outperforms the mathematical model both in respect to solution quality and computation time. We also show that there are significant cost savings by integrating these problems rather than solving them in a hierarchical manner.

The remainder of the paper is organized as follows. Section 4.2 briefly presents relevant literature. Section 4.3 includes the problem definition. Section 4.4 provides the mathematical model and enhancements on this formulation, while Section 4.5 presents a new method to obtain lower bounds. The heuristic is detailed in Section 4.6. The results are discussed in Section 4.7 and finally, the conclusions and future research perspectives are presented in the last section.

4.2 Relevant literature

The problem studied in this paper is related to the ship loading operations, and it covers aspects such as stowage planning, load sequencing, and handling equipment routing and scheduling. A detailed literature review on all of these components can be found in Iris and Pacino (2015).

The stowage planning problem has been addressed in two different ways in the literature. There are papers that aim at minimizing handling costs ensuring stability and seaworthiness of a ship in its route containing multiple ports. These studies agree that the problem belongs to the liner shipping company (see Pacino et al. (2011), Parreno et al. (2016)). There are also papers that study a variant of the problem at a single container terminal. We first review these studies in the second category. Imai et al. (2002) is one of the first papers that addresses the stowage planning at a single terminal with the aim of minimizing yard re-handles and the stability measure GM (i.e. the distance between the center of gravity and the metacenter). Later, Imai et al. (2006) include trim and healing to the objective function, and they also extend the problem by covering multiple rows in the yard. In Ambrosino and Sciomachen (2003), a stowage planning problem is solved in the first stage, then two yard-handling strategies are evaluated with the suggested stowage plan. Vessel stability is reflected by balancing the front-back and right-left side of the ship (the details of the stowage planning problem are in Ambrosino et al. (2004)). None of these studies considers the planning of the yard equipment that transfer containers to the ship, and the sequencing of the loading. Steenken et al. (2001) solve the stowage planing problem for a single terminal, and they integrate the problem with the assignment and scheduling of the Straddle Carriers (SCs) for the loading operations. The authors approach to the problem with a just-in-time method, they solve a model that assigns each container to a specific position and a specific SC. The

model is resolved when enough containers accumulate in the yard. Recently, Monaco et al. (2014) distinguish between the stowage planning problem solved by the liner shipping company (resulting in a class-based stowage plan) and the specific container assignment problem of the terminal (called *operational stowage planning problem*). They solely consider the operational stowage planning problem, and solve it through a two-phase tabu search method.

The second component of the FSLP is the load sequencing problem in which the loading order of containers is decided. Such a problem is directly attached to QC assignment and scheduling problem for a single ship (See examples such as Kim and Park (2004), Legato et al. (2012)) where required QCs are assigned to each set of bays, and the loading order of the bays and positions are determined for each QC. The load sequencing problem, which also determines the retrieval order from the yard, is also related to the locations of the containers in the yard (See papers that determine yard location, e.g. Jiang and Jin (2017)). Ji et al. (2015) address the load sequencing problem for loading a ship with multiple QCs and yard configurations. The authors suggest three different container relocation strategies which determine the relocation position of the blocking containers in the yard. Authors solve the problem through a Genetic Algorithm (GA). Bian et al. (2016) determine the loading sequence considering the number of re-handles in the yard. Authors assume that a detailed stowage plan is given, and one QC is available to load the ship. They suggest a two-phase method where they first order containers which do not require any re-handling. Then, they use a dynamic programming algorithm to sequence the remaining containers. Kim et al. (2004) integrate the load sequencing and Transfer Crane (TC) scheduling considering a given class-based stowage plan. The vessel stability is ensured by imposing weight and height limit constraints on the stacks. The load sequencing is optimised, but a column-wise loading policy is prioritized. A two-stage method is suggested. In the first stage, yard-clusters are sequenced (as in Lee et al. (2005)), while specific containers are sequenced in the second stage. There are also papers that address the integrated load sequencing and QC planning methods including internal reshuffles within bay of the vessel (e.g. Meisel and Wichmann (2010), Ding et al. (2017)).

Carlo et al. (2014b) point out that there are many papers about the assignment of the transport equipment to QCs and/or containers (e.g. Bish et al. (2005)), and the scheduling and routing of these equipment (e.g. Kim and Kim (1999), Zeng and Yang (2009)) during loading operations. We solely review studies that integrate the handling equipment planning with load sequencing and/or stowage planning. Alvarez (2006) integrate reach-stackers scheduling with stowage planning. The paper assumes that loading policy is either column-wise or layer-by-layer. The problem aims at minimizing the number of re-handling, the traveling distance and vessel instability. A Tabu Search (TS) based solution method is suggested. In a later work (Alvarez (2008)), a Lagrangian Relaxation (LR) based solution approach has been suggested for a similar problem. Jung and Kim (2006) integrate the load scheduling with equipment assignment (which is yard crane in this case) problems for a single ship. The authors study the interference between two yard cranes with the objective of minimizing makespan.

4.3 The Flexible Ship Loading Problem

In Figure 4.1, we illustrate the FSLP as the integration of four planning problems in the terminal, namely operational stowage planning, load sequencing, equipment assignment and equipment

scheduling. This problem is configured to load a single ship, but it could be extended for multiple ships.

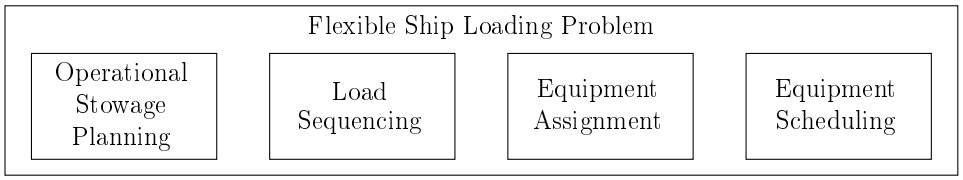


Figure 4.1: Flexible Ship Loading Problem composition

We now detail the FSLP by describing each of its components and their interactions. The operative stowage planning problem (Monaco et al. (2014)) deals with assigning specific containers in the yard to one slot (i.e. position) on the vessel with respect to the class-based stowage plan supplied by liner shipping company. In Figure 4.2, a class-based stowage plan is illustrated for a bay of a ship. In this figure, there are four container classes to be loaded into the bay, namely A, C, D, E, and the positions of containers in the yard area are shown. The operative stowage plan points out the position for each specific container obeying the class-based stowage plan. Figure 4.2 is a simplified example for one bay and yard, while the problem is solved for all bays that will be loaded.

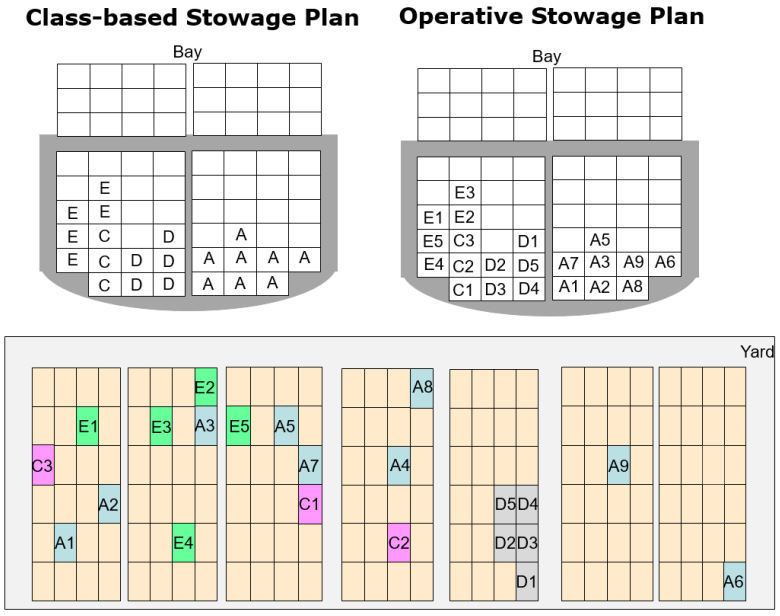


Figure 4.2: Class-based stowage plan to operative stowage plan

The flexibility of selecting the specific container (among the same class containers) can bring significant savings on the traveling distance, and consequently the traveling time, of all containers. Let us consider the example in Figure 4.3 with two containers of the same class (say two 40-foot containers weighting between 20 and 22 tonnes having the same destination) are to be loaded. The position of each container in the yard area is also shown, and an arrow represents the traveling distance needed to bring the container to the vessel. In the figure on the right side, the terminal has made a better operative stowage plan that requires less traveling distance compared to the operative stowage plan on the left side.

The flexible assignment of containers in the same class helps to integrate the remaining problems into the FSLP. The operative stowage plan is of no consequence for the liner so long as the container classes are not changed.

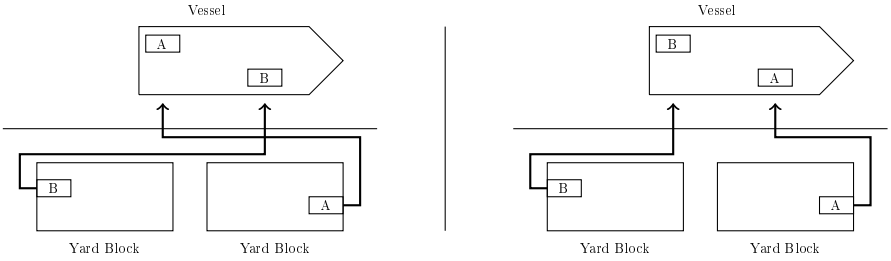


Figure 4.3: Different operative stowage plans: Effect on travel distance.

The second component of the FSLP is the load sequencing problem which determines the loading sequence of the containers in the yard. The sequence in which containers will be loaded is governed by physical rules, and due to the specific layout of the vessel some positions must be loaded before others. The sequence is affected by the ready time (i.e. the time a container is in front of the respective QCs) of each container, and this depends on various factors such as container locations in the yard, the availability of TVs, the operative stowage plan, etc. The terminal might reduce the total loading time by efficiently sequencing containers (See Kim et al. (2004)).

The load sequencing problem is constrained by the QCs work-schedule. The QCs work-schedule is a set of decisions that includes the QCs assignment to bays of the vessel and the loading order between the bays. The QC work-schedule is mostly determined in earlier stages with berth allocation and QC assignment problem (See Iris et al. (2015), Turkogullari et al. (2016), Iris et al. (2017)).

The FSLP finally covers the assignment and scheduling of transport equipment (i.e. transfer vehicles). Integrating the TVs into the FSLP is vital because they are limited resources in the terminal, and they might cause a bottleneck during loading operations. Moreover, generating a feasible schedule of TVs will determine the ready time of each container in front of the respective QC more accurately. These ready times influence the assignments of containers to each position (operative stowage plan). In this study, time is discretized by minutes, and the time unit is one minute. The FSLP studied in this paper covers the assignment of specific TVs to each QC and the scheduling of all TVs to load all containers to the vessel. In other words, the problem deals with determining which specific container will be picked up by which TV at what point in the time. It is very common that the number of TVs that works on each QC change over time (time-variant TV assignment). This means that for example, a solution can hold 3 TVs working on a QC for some time then it can be reduced to 2 TVs for the remaining loading time. We consider such a time-variant TV assignment in this study.

The problem definition is based on the following assumptions:

- Unloading operations are performed first, then loading operations start (i.e. the problem does not include dual cycling (Goodchild and Daganzo (2007))).
- The work-schedule for each QC is determined beforehand. This also means that the se-

quence, in which each position is loaded by each QC, is known. As the load sequencing policy, "from-sea-to-land" with "stack-wise" sequencing is applied to each QC.

- The retrieval order of containers in the yard does not generate any yard-shifts within the yard bay (e.g. a variant of the pre-marshalling policy).
- The stability of the vessel is ensured with the class-based stowage-plan.
- Each TV can only work for a single QC during the loading of the vessel. In other words, it is not allowed to pool TVs for QCs, and all TVs are identical.
- TV operations are non-preemptive. When a TV is assigned to a QC it does not stop until it finishes the given tasks on that QC, and each TV is in front of its respective QC in the initial position.
- The congestion in the yard, the travel speed of a TV with/without a container are all reflected in the transportation times between yard positions (or Input/Output points depending on the yard layout type) and QCs.
- There is no container buffer area under the QC, and this means the TV and QC operations are not decoupled.

A feasible solution to the FSLP holds the assignment of each container to a vessel position. A solution also shows which TV will transfer each container to its position, the time of pickup from the yard block and delivery time in front of the QC. The complete schedule of all TVs is also made. A feasible solution fulfills the requirements of the class-based stowage plan, QC work-schedules and the load sequencing policy. We believe that our study is the first to integrate the above four problems. It utilizes the collaboration between liner shipping company and terminal operators.

4.4 Mathematical Model for the FSLP

The list of notations, i.e. parameters, decision variables, is as follows:

Parameters and sets:	
C	Set of containers that will be loaded to vessel
C_p	Set of containers belonging to a class suitable for slot position p
Q	Set of quay cranes that are assigned to load the vessel
Q_p	Set of quay crane that loads position p (one element set)
P	Set of positions to be loaded
P_i	Set of positions that match with the class type of container i
P_q	Set of positions that will be loaded by QC q
P_p^{crane}	Set of positions that are handled by the same crane as position p
S	Set of transfer vehicles available to serve the vessel
S_p	Set of transfer vehicles that are available to serve position p
S_q	Set of transfer vehicles that are available to serve QC q, $S_q \in \left\{ s_q^1, s_q^2, .., s_q^{\lvert S_q \rvert} \right\}$

T	Set of time periods, $T \in \{0, 1, .., H-1\}$, where H is the closing of planning horizon
τ_{ip}	Time needed for a transfer vehicles to transport container i from its yard-position to vessel-position p. The time needed is assumed to be equal in both directions.
β	The loading time for each QC, minimum time between two consecutive container loading operation
EFT	Expected finishing time of operations for the vessel
α	The cost of using one TV for one time-unit
γ	The cost of exceeding the expected finishing time (EFT) for one time-unit
M	A large positive number
	Decision variables:
$t_p^s \in \mathbb{Z}^+$	Time when container for position p has been dropped in front of QC by TV s (container dropping time)
$Start_s \in \mathbb{Z}^+$	Time when operations of TV s starts
$End_s \in \mathbb{Z}^+$	Time when operations of TV s ends
$z \in \mathbb{Z}^+$	Makespan for the loading of entire ship (operations)
$\Delta EFT \in \mathbb{Z}^+$	Lateness of operations
$x_{ip}^s \in \mathbb{B}$	1; if the container i is loaded to position p, and it is picked up by TV s, 0 otherwise

As per the second assumption mentioned in Section 4.3, the ordering of the positions is determined beforehand. We use the notation $p' \prec\prec p$ to indicate that the position p' is handled immediately before position p according to the ordering, while $p' \prec p$ indicates that position p' is loaded before position p, by the same QC.

The binary decision variables x_{ip}^s correspond to the operative stowage plan and TV assignment to containers, while integer variables $t_p^s, Start_s, End_s$ handle the TV scheduling. Let us now introduce the mathematical model:

$$\min \alpha \sum_{s \in S} (End_s - Start_s) + \gamma \Delta EFT \tag{4.1}$$

subject to

$$\sum_{p \in P_i} \sum_{s \in S_p} x_{ip}^s = 1 \qquad \forall i \in C \tag{4.2}$$

$$\sum_{i \in C_p} \sum_{s \in S_p} x_{ip}^s = 1 \qquad \forall p \in P \tag{4.3}$$

$$2\tau_{ip} - M(2 - x_{ip}^s - \sum_{i\prime \in C_{p\prime}} x_{i\prime p\prime}^s) \leq t_p^s - t_{p\prime}^s \qquad \forall i \in C, \forall s \in S, \forall p \in P_i, p\prime \in P_p^{crane} \mid p\prime \prec p \tag{4.4}$$

$$\sum_{s \in S} t_p^s \geq \sum_{s \in S} t_{p\prime}^s + \beta \qquad \forall p \in P, p\prime \in P_p^{crane} \mid p\prime \prec\prec p \tag{4.5}$$

$$t_p^s \leq \sum_{i \in C_p} H x_{ip}^s \qquad \forall p \in P, \forall s \in S_p \tag{4.6}$$

$$t_p^s \geq 2 \sum_{i \in C} \tau_{ip} x_{ip}^s \qquad \forall p \in P, \forall s \in S_p \tag{4.7}$$

$$t_p^s - 2 \sum_{i \in C} \tau_{ip} x_{ip}^s + H(1 - \sum_{i \in C} x_{ip}^s) \geq Start_s \qquad \forall s \in S, \forall p \in P \tag{4.8}$$

$$t_p^s \leq End_s \qquad \forall s \in S, \forall p \in P \tag{4.9}$$

$$Start_s \leq End_s \qquad \forall s \in S \tag{4.10}$$

$$z \geq t_p^s + \beta \qquad \forall s \in S, \forall p \in P \tag{4.11}$$

$$x_{ip}^s = 0 \qquad \forall i \in C, \forall p \in P, \forall s \in S \setminus S_p \tag{4.12}$$

$$\Delta EFT \geq z - EFT \tag{4.13}$$

$$t_p^s, Start_s, End_s, z, \Delta EFT \in \{0, .., H-1\} \qquad \forall s \in S, \forall p \in P \tag{4.14}$$

$$x_{ip}^s \in \{0, 1\} \qquad \forall i \in C, \forall s \in S, \forall p \in P \tag{4.15}$$

The objective function (4.1) is a combination of the cost of service times of TVs and the lateness (if the ship loading finishes after the expected finishing time). Constraint (4.2) ensures that each container will be loaded to a position that matches with its container class. Constraint (4.3) guarantees that all positions are loaded with a container that matches the container class of that position. For a given container, TV and position, constraint (4.4) makes sure that the container dropping time for that position is set correctly. This is done by forcing the difference between two consecutive positions‘ dropping times to be greater than or equal to the time required to bring the container ($2\tau_{ip}$) in front of the QC. The term multiplied by M on the left-hand side in constraint (4.4) makes sure that the constraint is only active when the two positions which are following each other in loading sequence are transported by the same TV. Constraint (4.5) ensures that all positions are loaded in the correct order, and the containers that will arrive at the same QC should have at least β time apart. Constraint (4.6) ensures that the container dropping time is earlier than the end of the planning horizon. Constraint (4.7) guarantees that the earliest dropping time for a position is the transportation time of the container which is loaded to that position. Constraint (4.8) sets the starting time of each TV operation, while constraint (4.9) sets the ending time of each TV operation. If a TV is not assigned to any QCs, these variables take a value of zero. Constraint (4.10) is the link between the starting and ending time for each TV operation. Constraint (4.11) obtains the makespan, while constraint (4.12) ensures that a container for position p cannot be picked up by TV s if TV s is not assigned to serve the position. Constraints (4.14)-(4.15) determine the domain of variables. Firstly, we formulate an upper bound on H simply by assuming that only one QC and one TV are used. We can, thus, obtain an upper bound on the planning horizon $H = \max\Big\{\sum_{p \in P}\Big\{2 \max_{c \in C_p}\{\tau_{cp}\} + \max\big\{0, (\beta - 2 \min_{c \in C_p}\{\tau_{cp}\})\big\}\Big\}, \beta|P|\Big\}$.

4.4.1 Enhancements for the FSLP model

This section introduces enhancements for our formulation. The enhancements are based on formulating lower bounds on variables and valid inequalities for the FSLP.

4.4.1.1 Lower bounds on the variables

Let us first formulate the minimum total transportation time that is required to transport containers that will be loaded by QC q. The minimum overall time needed to transport all containers of a particular QC q (δ^q_{min}) is obtained by solving an assignment problem that minimizes the total required transportation time. Let x_{cp} be the assignment variable, i.e. $x_{cp} = 1$ if container $c \in C$ is assigned position $p \in P$, and 0 otherwise. The value of δ^q_{min} can be calculated by solving the model in (4.16). The objective function minimizes the transportation time obeying the class-based stowage plan, and the first constraint ensures that all positions that will be loaded by QC q must get exactly one container, and the second constraint ensures that each container can be loaded at most one position.

$$\delta^q_{min} = \min \left\{ \sum_{c \in C} \sum_{p \in P_c} 2\tau_{cp} x_{cp} : \sum_{c \in C_p} x_{cp} = 1 \ \ \forall p \in P_q, \sum_{p \in P_c} x_{cp} \leq 1 \ \ \forall c \in C \right\} \qquad \forall q \in Q \quad (4.16)$$

We now set a lower bound on the completion time (i.e. makespan) variable z in constraint (4.17). The makespan should be larger than the maximum of the finishing times of all QCs. The lower bound on the finishing time of each QC is obtained by taking the maximum between the total loading time for QC q and the minimum transportation time needed to load all containers of QC q.

$$z \geq \max_{q \in Q} \left\{ \max\{\beta |P_q|, \left\lceil \frac{\delta^q_{min}}{|S_q|} \right\rceil \right\} \quad (4.17)$$

4.4.1.2 Valid inequalities for the FSLP model

We can formulate a better link between t^s_p and x^s_{cp} variables in constraint (4.18). For each position p, the sum of dropping times (t^s_p) for all TVs is at least the maximum of the total loading time of all positions before p and the minimum transportation time required to load all positions before p.

$$\sum_{s \in S_p} t^s_p \geq \max \left\{ \sum_{p' \prec p} \beta, \frac{\sum_{p' \prec p} \sum_{c \in C_{p'}} \sum_{s \in S_{p'}} 2\tau_{cp'} x^s_{cp'}}{|S_{Q_p}|} \right\} \quad \forall p \in P \quad (4.18)$$

In constraint (4.18), the total loading time for all positions loaded by same QC before p is $\sum_{p' \prec p} \beta$. To calculate the minimum time to transport all containers before p, we first sum all transportation times for positions before p ($\sum_{p' \prec p} \sum_{c \in C_{p'}} \sum_{s \in S_{p'}} 2\tau_{cp'} x^s_{cp'}$), we then divide this by total number of TVs assigned to that QC. This results in minimum transportation time for all positions before p. The two parts in constraint (4.18) are split up into two sets of constraints, to ensure the linearity of the model.

The next set of valid inequalities focuses on the container classes rather than specific containers. Let us name the set of container classes as U and the set of container classes for each QC q as

U_q. These sets can be easily obtained as we know the QC work-schedules and the class-based stowage plan. We, also, can obtain the set of containers which is in the class of u (C_u), and the set of positions which requires a container of class u that will be loaded by QC q (P_{qu}). Valid inequality (4.19) ensures that for each QC q and container class u belonging to set U_q, the total number of containers of class u to be loaded by QC q equals to $|P_{qu}|$.

$$\sum_{s\in S_q}\sum_{i\in C_u}\sum_{p\in P_{qu}} x_{ip}^s = |P_{qu}| \quad \forall q \in Q, \forall u \in U_q \tag{4.19}$$

We also formulate inequalities to break the symmetry. In this paper, it is assumed that all TVs available for a QC are identical, hence a generated pickup order for a TV is feasible for all TVs. This property generates many symmetrical solutions where there are a lot of exactly indifferent alternative TV assignments. For each QC, constraint (4.20) ensures that TVs are assigned with a lexicographical order in each symmetry class for all TVs except the highest indexed one.

$$End_s - Start_s \geq End_{s+1} - Start_{s+1} \quad \forall q \in Q, \forall s \in S_q \setminus \{s_q^{|S_q|}\} \tag{4.20}$$

Finally, we formulate a valid inequality that better links the assignment (x_{cp}^s) and scheduling variables ($End_s, Start_s$). Constraint (4.21) ensures that the service time of TV s ($End_s - Start_s$) is larger than the total transportation time to load containers that are going to be handled by that TV s.

$$End_s - Start_s \geq 2\sum_{i\in C_p}\sum_{p\in P_i} \tau_{ip} x_{ip}^s \quad \forall s \in S \tag{4.21}$$

We call the enhanced version of the FSLP model as FSLP+.

4.5 New lower bounds for the FSLP

To obtain new lower bounds for the FSLP, we focus on the components of the objective function, which are the cost of TV service times and the cost of ending later than expected finishing time. We formulate a new mathematical model that omits decision variables related to TV scheduling ($t_p^s, Start_s, End_s$), and this model obtains a lower bound for the FSLP. Let us first show that we can obtain lower bounds on each objective component by solely using x_{ip}^s variables.

Proposition 1: $\sum_{i\in C}\sum_{p\in P_i}\sum_{s\in S_p} 2\tau_{ip}x_{ip}^s$ is a lower bound on $\sum_{s\in S}(End_s - Start_s)$.

Proof: $\sum_{i\in C}\sum_{p\in P_i}\sum_{s\in S_p} 2\tau_{ip}x_{ip}^s$ constitutes the total transportation time of all TVs, while $\sum_{s\in S}(End_s - Start_s)$ is the service time, and it includes the total transportation time and the TV waiting times. Then, $\sum_{i\in C}\sum_{p\in P_i}\sum_{s\in S_p} 2\tau_{ip}x_{ip}^s \leq \sum_{s\in S}(End_s - Start_s)$ as the waiting time is always non-negative. □

Proposition 2: $\beta + \max_{s \in S} \left\{ \sum_{i \in C} \sum_{p \in P_i} 2\tau_{ip} x_{ip}^s \right\}$ is a lower bound on z.

Proof: Makespan (z) is bounded by the maximum of the finishing times of all TVs plus the loading time (β) of the last container. Then, we have to show that $\sum_{i \in C} \sum_{p \in P_i} 2\tau_{ip} x_{ip}^s$ is a lower bound on the maximum finishing time of each TV. The finishing time of TV s is at least the summation of all transport times for containers that it will load by that TV. □

Proposition 3: $\max_{q \in Q} \left\{ \beta |P_q| \right\}$ is a lower bound on z.

Proof: Makespan (z) is bounded by the maximum of the finishing times of all QCs that work on the vessel. The finishing time of one QC is at least loading time of all positions that the QC is assigned to ($\beta |P_q|$). So that, the maximum of all total loading times is a lower bound on z.□

We now suggest a mathematical model to obtain the lower bound on the FSLP using above propositions, the model uses the same notation and variables of the FSLP model. We introduce a new integer variable, TTS_s, which presents the lower bound on the finishing time of operations for TV s. Now, let us introduce the new model which is called lower bound model, and it is abbreviated as LB-FSLP:

$$\min \alpha \sum_{i \in C} \sum_{p \in P_i} \sum_{s \in S_p} 2\tau_{ip} x_{ip}^s + \gamma \Delta EFT \tag{4.22}$$

subject to

$$\sum_{p \in P_i} \sum_{s \in S_p} x_{ip}^s = 1 \qquad \forall i \in C \tag{4.23}$$

$$\sum_{i \in C_p} \sum_{s \in S_p} x_{ip}^s = 1 \qquad \forall p \in P \tag{4.24}$$

$$x_{ip}^s = 0 \qquad \forall i \in C, \forall p \in P, \forall s \in S \setminus S_p \tag{4.25}$$

$$TTS_s = \beta + \sum_{i \in C} \sum_{p \in P_i} 2\tau_{ip} x_{ip}^s \qquad \forall s \in S \tag{4.26}$$

$$z \geq TTS_s \qquad \forall s \in S \tag{4.27}$$

$$z \geq \beta |P_q| \qquad \forall q \in Q \tag{4.28}$$

$$z \geq \left\lceil \frac{\delta_{min}^q}{|S_q|} \right\rceil \qquad \forall q \in Q \tag{4.29}$$

$$\Delta EFT \geq z - EFT \tag{4.30}$$

$$TTS_s, TTQ_q, z, \Delta EFT \in \{0, .., H-1\} \qquad \forall s \in S \tag{4.31}$$

$$x_{ip}^s \in \{0, 1\} \qquad \forall i \in C, \forall s \in S, \forall p \in P \tag{4.32}$$

The optimal solution to (4.22)-(4.32) is a lower bound on the FSLP. The objective function (4.22) is a combination of the cost of TV transportation times and the cost of lateness. Constraints

(4.23)-(4.25) are interpreted in a similar way as with constraints (4.2), (4.3) and (4.12) of the FSLP model. Constraint (4.26) sets the lower bound on the finishing time for each TV, constraint (4.27) uses these variables to obtain a lower bound on the makespan. Constraint (4.28) sets the lower bound on finishing time for each QC, where $|P_q|$ refers to the number of positions that will be loaded by QC q. Constraint (4.29) uses the minimum transportation time to obtain the lower bound on the makespan for the vessel. Constraints (4.31)-(4.32) define the domains of variables.

4.6 Heuristic approach

Broadly speaking there are two main decisions to be taken in the FSLP. 1) The service times for the TVs where $End_s - Start_s$ is service time for TV s and 2) the container-assignment for each TV (x^s_{ip}).

The objective function heavily depends on the service times of the TVs, which on the other hand depend on the container-assignment. We propose a Greedy Randomised Adaptive Search Procedure (GRASP) which initially imposes a specific service time to each TV. The scheduling and container-assignment are then created attempting to respect the assigned service times. Within each GRASP iteration, the generated solution is evaluated. Should a solution be promising enough, it is improved using a local search method.

4.6.1 Construction heuristic

The heuristic is based on the assumption that it is not important when a TV works, what is important is the duration i.e. the length of the service window. We do not need to distinguish between the different characteristics of the TVs, as they are all assumed identical. Given an arbitrary assigned service time for each of the TV $s \in S_q$ of QC $q \in Q$ (defined by $\overline{Start}_s$ and $\overline{End}_s$), a solution can be constructed with the following four steps:

Step 0: Select QC
The construction heuristic builds the solution by considering the QCs in a sequential order. The first step is thus to select the next QC q to build the solution for.

Step 1: Assign containers to positions
Let $\langle c, p\rangle$ be the assignment of container $c \in C_p$ to position $p \in P_q$, and $\Upsilon(q)$ be the set of all container-assignments for the positions serviced by QC $q \in Q$. Starting from the first position to be loaded until the last, the assignment of a container is done greedily by selecting the closest available container. Thus, for an arbitrary position p we select container

$$c = \underset{c' \in C_p(x)}{\arg\min} \left(2\tau_{c'p}\right), \tag{4.33}$$

where $C_p(x)$ is the set of compatible containers for position p that have not yet been assigned in solution x.

Step 2: Assigning TVs to positions
Let $\langle c^*, p^* \rangle \in \Upsilon(q)$ describe the next container-assignment to which a TV needs to be dispatched (initially defined as the container-assignment for the first position in the loading order of QC q).

For the position p^* let $\Upsilon(q, p^*)$ be the set of the next ι container-assignments including $\langle c^*, p^* \rangle$. Here ι is a parameter controlling the size of the set $\Upsilon(q, p^*)$. Also, let S^A be the set of *active* TVs $s \in S_q$. A TV is said to be active if it can service the container-assignment $\langle c^*, p^* \rangle$ without exceeding the assigned end time ($\overline{End}_s$). More formally a TV is active if $\max(a_s + 2\tau_{c^*p^*}, d_{p-1} + \beta) \leq \overline{End}_s$, where a_s is the time TV s is available (i.e. the time at which it finished its last container delivery, or if no deliveries have been assigned $\overline{Start}_s$), and where d_{p-1} is the time at which the QC began to service the previous position. If the QC has not served any position then $d_{p-1} = -\beta$. Should $S^A = \emptyset$ then we will consider $S^A = S_q$.

We now select, through complete enumeration, the sequence of active TVs to service each container-assignment in $\Upsilon(q, p^*)$ which results in the minimum *completion time*. In this context, the completion time is the time in which QC q has finished loading the last container-assignment of $\Upsilon(q, p^*)$. For few TVs and small ι partitions, the exponential growth of the number of sequence combinations is not an issue for the complete enumeration. We only consider the active TVs to prioritise the scheduling of TVs that do not exceed the EFT. This is due to the way TV service times are generated (more details are provided in Section 4.6.2.2).

Step 3: Assign TV and update times
Given the TV sequence found in Step 2, we only commit to the solution the TV scheduled for container assignment $\langle c^*, p^* \rangle$. Practically we are only assigning the TV to the very next position to load (p^*). Hereafter, the selected TV available time and the QC time are updated (a_s and d_p). The container-assignment $\langle c^*, p^* \rangle$ is then removed from $\Upsilon(q)$. Should $\Upsilon(q) = \emptyset$, go to Step 0 and process the next QC, otherwise go to Step 2.

4.6.2 GRASP

GRASP (Feo and Resende, 1989; Feo and Resende, 1995), is a multi-start iterative metaheuristic that combines a constructive phase with an improvement phase. In each iteration, a solution is built from scratch using a randomised construction heuristic, where a random strategy is used to provide diversity. The improvement phase consists of a local search method used to improve the solution found in the constructive phase. The GRASP method has successfully been applied to many optimisation problems, such as the container stowage slot planning problem (Parreno et al., 2016), Vehicle routing problems (Kontoravdis and Bard, 1995) and the quadratic assignment problem (Pardalos and Resende, 1994), among others. A general outline of the GRASP algorithm for the FSLP is presented in Algorithm 3.

The algorithm begins by generating a solution using a randomised version of the construction heuristic described earlier (line 3). The randomised construction heuristic is described in Section 4.6.2.1.

After a feasible solution has been found, it is then passed to the improvement phase. Two improvement heuristics have been implemented, each focusing on different aspects of the solution

Algorithm 3 GRASP

```
1: x^b ← ∅
2: while not Terminate() do
3:    x ← RandomisedConstructionHeuristic()
4:    x ← VehicleReassignment(x)
5:    if f(x) < f(x^b)(1 + κ) then
6:       x ← ContainerSwap(x)
7:    end if
8:    UpdateBest(x^b, x)
9: end while
10: return x^b
```

and thus complementing each other. The first method is aimed at improving the TVs' schedule, and it is used at every iteration (line 4). The second focuses on the container assignment, however, due to its computational complexity, it is only used when the cost of the candidate solution ($f(x)$) is within κ percentage of the best-found solution cost ($f(x^b)$). The improvement methods are described in details in Section 4.6.2.3 and Section 4.6.2.4 respectively.

In each iteration, a new solution is made from scratch, and if the newly generated solution is better than the previous best, it is kept (line 8). Once the termination criterion is reached, the algorithm returns the best-found solution. In our approach, we use the number of iterations as the termination criterion. Following we describe the implementation of each GRASP component in detail.

4.6.2.1 Randomised construction heuristic

Randomization is included into the construction heuristic (Section 4.6.1) in three places: the order in which QCs are processed, the container to position assignment (Step 1), and the generation of the service times, which the heuristic is based on.

Randomizing the order in which the QCs are processed (Step 0), results in variations on the assignment of containers to position. For the actual container to position assignment, let ρ_c be a random number in the interval $[0.5; 1.5]$. At each iteration, a random number ρ_c is sampled for every container c. The container c that minimises the driving distance times ρ_c is assigned to the position p, thus effectively changing eq. (4.33) to

$$c = \underset{c' \in C_p(x)}{\arg\min} \left(\rho_{c'} 2\tau_{c'p}\right) \qquad \forall p \in P_q$$

The last source of randomisation, the generation of service times, also takes care of the adaptive part of the procedure (as explained in Section 4.6.2.2). Service times are generated on a per QC basis using the following concept. Consider a QC $q \in Q$. We define the total service time to be $\nu_q = \sum_{s \in S_q} \overline{End}_s - \overline{Start}_s$. Since the TVs are operating in parallel, the largest total service time that a QC can have, without any delays, is $|S_q|EFT$. This corresponds to all TVs starting at time 0 and ending at time EFT. The value of ν_q can be used to guide the generation of the service time of each TV. We do so by imposing $\overline{Start}_s = 0$ for all TVs $s \in S_q$. We then assign $\overline{End}_s = EFT$ for as many TVs as possible. Hence exactly $\left\lfloor \frac{\nu_q}{EFT} \right\rfloor$ TVs will have this assignment

for QC q. Any residual value $\nu_q - \left\lfloor \frac{\nu_q}{EFT} \right\rfloor$ will be assigned to one TV, while all remaining TVs are assigned $\overline{End}_s = 0$. The service times assignment is the base of the construction heuristic. Consider now Step 2 of the heuristic. The procedure starts by scheduling only the active TVs, i.e. the TVs $s \in S_q$ that can finish the next scheduled container-assignment ($\langle c^*, p^* \rangle$) within $\overline{End}_s$. By doing so, we make sure that no TV is scheduled after the $\overline{End}_s$ unnecessarily, which also conforms to the way the service times are generated. Notice, however, that should this not be possible the heuristic will reset the set of active TVs to be equal to the complete set of TVs (S_q) thus allowing scheduling beyond $\overline{End}_s$.

The randomization of the service time generation is rooted on the selection of the total service time of each QC (ν_q). At each iteration the value of ν_q is selected at random within the range $[EFT; |S_q|EFT]$. To better guide the selection of ν_q we partition the range into intervals of size ε, effectively generating the following set of intervals, I_q:

$$\{[EFT; EFT + \varepsilon], [EFT + \varepsilon; EFT + 2\varepsilon], \ldots, [|S_q|EFT - \varepsilon; |S_q|EFT]\} \cup$$
$$\{[EFT; EFT], [|S_q|EFT; |S_q|EFT]\}$$

The values of ν_q are then selected at random within one of these range intervals. When the expected finishing time is low, it would most likely be best to assign all available TVs. On the other hand, when EFT is high, one TV operating is likely to be cost optimal. For these reasons, we have also included the ranges $[EFT; EFT]$ and $[|S_q|EFT; |S_q|EFT]$ in the set. We call these range intervals Service Time Intervals (STIs).

4.6.2.2 Adaptivity

When selecting the value ν_q, a STI (i) is first chosen, and ν_q is selected at uniform within the range interval described by i. The adaptivity of the GRASP method comes from how the probability of choosing a STI adaptively changes throughout the execution of the algorithm. For each STI, let $\mathbb{P}_{iq}$ be the probability of choosing STI i for QC q, calculated using the roulette wheel selection principle

$$\mathbb{P}_{iq} = \frac{w_i}{\sum_{j \in I_q} w_j} \qquad \forall q \in Q, i \in I_q \tag{4.34}$$

Where w_i are the weights. This probability is adaptively adjusted in a similar manner as in the ALNS method described in Ropke and Pisinger (2006).

Throughout the algorithm, we keep track of the best-found solution and its value $\widehat{z}$. The positions to be loaded by QC q are pre-determined, thus we also keep track of the best partial solution scheduling container-assignments for QC q. The cost of a partial QC solution is calculated as

$$\widehat{z_q} = \alpha \sum_{s \in S_q} (End_s - Start_s) + \gamma \max\left(\left(\max_{s \in S_q} End_s\right) + \beta - EFT, 0\right) \tag{4.35}$$

Where $\max\left(\left(\max_{s \in S_q} End_s\right) + \beta - EFT, 0\right)$ is the tardiness of the operation for QC q.

The execution of the algorithm is divided into a number of *segments* i.e. a number of η consecutive iterations. The score obtained by STI i in segment j (denoted π_{ij}) is updated according to the

following three parameters; σ_1, σ_2 and σ_3. If a new best solution is found, σ_1 is added to the score for all the chosen STIs that contributed to finding this solution. If the cost for QC q (z_q) is better than the previously best solution for that QC, $\widehat{z_q}$, σ_2 is added to the score for the STI chosen for QC q. Last, if for a QC q the cost (z_q) is within δ percentage of the best for QC q ($\widehat{z_q}$) then σ_3 is added to the score for the chosen STI for QC q.

After η iterations are executed the weights are updated as follows.

$$w_{ij+1} = w_{ij}(1-r) + r\frac{\pi_{ij}}{\theta_{ij}} \qquad \forall q \in Q, i \in I_q$$

Here w_{ij} is the weight for STI i in segment j, θ_{ij} is the number of times STI i was chosen in segment j and r is the *reaction factor*. The reaction factor controls how quickly the weight adjustment reacts to changes in the effectiveness of each STI.

The adaptivity as described will make the STIs which contributes to finding good solutions more probable. Furthermore, it makes the overall heuristic robust towards different characteristics in the problem that have an impact on the time needed to load the containers for a QC.

4.6.2.3 Vehicle reassignment

The first improvement method looks at the TV scheduling for a QC q. The TV scheduling described in Section 4.6.1 Step 3 is myopic at best, and only aims at minimising tardiness. It does not consider TV waiting time and the minimization of the total service time.

The TV reassignment procedure is aimed at reducing the service times of each TV. Since the service time is dictated by the start ($Start_s$) and end (End_s) of a TV operation, for a given TV $s \in S_q$, we consider the possibility of re-assigning its first or last container to a different TV $s' \in S_q \setminus \{s\}$. For a given QC q we generate all the possible re-assignments of containers to TVs (which is at most $2|S_q|-1$ since we only consider two containers per TV). The re-assignment that best improves the objective is then applied. The procedure restarts every time a new improving re-assignment is found until no new re-assignment is available and all the QCs are processed.

4.6.2.4 Container swapping

The second improvement method is a local search based on a swap neighbourhood operator. The aim of the local search is to find improvements in the container assignments. The neighbourhood is defined by all the possible container swaps within the same container class. Here a swap means that two positions exchange containers, and consequently the two TVs scheduled to the positions will change the container they pickup. The neighbourhood operator, however, only evaluates a limited number of swaps to reduce the number of evaluations. The most improving swap is then applied to the solution. The container swapping procedure is further described in Algorithm 4.

Given an input solution x, Algorithm 4 starts by initializing z^* and $Move$ which will hold the best evaluation and swapping move respectively (lines 3-4). The algorithm then proceeds to evaluate the possible swaps. For each class u, in the set of container classes U, we select $|C_u|$

Algorithm 4 `ContainerSwap`

Input: x
1: Terminate $\leftarrow$ **false**
2: **while not** Terminate **do**
3: $z^* \leftarrow 0$
4: $Move \leftarrow \emptyset$
5: **for all** $u \in U$ **do**
6: **for** $i = 1$ **to** $|C_u|$ **do**
7: $c_1 \leftarrow$ Random container $c \in C_u$
8: $c_2 \leftarrow$ Random container $c \in C_u \setminus \{c_1\}$
9: $z \leftarrow$ `EvaluateSwapImprovement`(x, c_1, c_2)
10: **if** $z > z^*$ **then**
11: $z^* \leftarrow z$
12: $Move \leftarrow \{c_1, c_2\}$
13: **end if**
14: $i \leftarrow i + 1$
15: **end for**
16: **end for**
17: **if** $z^* > 0$ **then**
18: $x \leftarrow$ `PerformMove`$(x, Move)$
19: **else**
20: Terminate $\leftarrow$ **true**
21: **end if**
22: **end while**
23: **return** x

random swaps, where C_u is the set of all containers of class u (lines 5-8). Each swap is evaluated (line 9) and, if it is improving, its value and move are stored in z^* and $Move$ (lines 10-13). The best improving swap is then selected and applied to the solution (lines 17-18). Should we not be able to find such a swap, the procedure terminates and returns the locally improved solution, (lines 2, 19-23)

Calculating a swap improvement in `EvaluateSwapImprovement`(x, c_1, c_2) is the most computationally expensive part of this method. In Section 4.A, we provide details of how caching techniques can be used to implement this operation efficiently.

4.7 Computational analysis

We now analyse the performance of each formulation, valid inequalities and the GRASP heuristic. All methods are executed using a 2.30 GHz Intel Xeon E5 Processor, and computational times are reported in seconds. All models are solved using CPLEX 12.7.0. A time limit of 3 hours is imposed to solve the models with the options of *emphasizing optimality*. In default conditions, models are run with four threads.

The GRASP heuristic have been implemented in `Java` 1.8, and have been tuned using the *Gender-Based Genetic Algorithm for the Automatic Configuration of Algorithms* (gga) described in Ansótegui and Sellmann (2009). Table 4.2 describes how we tuned the algorithm. The first two columns describe the parameters, first brief in text then the symbol used. The next two column contains information for the tuning, first which values we tested, then the start value we used. For some of the parameters we tested a discrete set of values, and for others we tested all

integers in a range (symbolised by [min; max]). For the tuner, additional instances have been generated, and the heuristic terminates with the best-found solution after 30 seconds. The value in the last column is the value that the tuner finds perform best.[1]

Table 4.2: Description of the parameter tuning

Description	Symbol	Test	Start	Tuned
STI length	ε	$\{0.01, 0.025, 0.05, 0.075, 0.1, 0.125, 0.15, 0.2\}$EFT	0.1EFT	0.15EFT
Vehicle assignment lookahead	ι	$\{1, 2, 3, 4, 5\}$	4	2
Container swapping use percentage	κ	$\{0, 0.05, 0.1, 0.15, 0.2, \infty\}$	0.1	0.2
Reaction factor	r	$\{0.025, 0.05, 0.075, 0.1, 0.125, 0.15, 0.2\}$	0.1	0.1
Segment size	η	$\{25, 50, 75, 100, 150, 200, 250, 300, 350, 400, 500\}$	200	25
STI score update: New best	σ_1	$[30; 75]$	50	45
STI score update: New crane best	σ_2	$[20; 50]$	33	36
STI score update: New crane solution within δ of crane best	σ_3	$[10; 45]$	19	17
The δ parameter associated with σ_3	δ	$\{0.01, 0.025, 0.05, 0.075, 0.1, 0.125, 0.15, 0.2\}$	0.05	0.2

4.7.1 Data description

To test the formulations and the heuristic a benchmark set has been generated. The number of containers to be loaded in the instances are either 60, 240, 500 or 1000, corresponding to small, medium and large vessels. To describe different yard structures, we test three different *densities*. The density will affect the travel time to the containers in the yard. The different densities are *Uniform*, *Scattered* and *Less Dense*. For the Uniform case, all containers are stored closely together in the yard, minimising the variance in the travel times. In the scattered case the containers are stored over a larger area, and it implies a higher variance in the travel times. The Less Dense is a mix between these two densities.

In total 30 instances has been generated. The number of container types ranges from 10 to 100. In the benchmark, there are 3 TVs assigned to a QC, and either 2 or 4 QCs are available depending on the size of the vessel. The QC loading time β is 1 in all of the instances. See Tables 4.3 and 4.4 for an overview of the instance characteristics.

The expected finishing time (EFT) corresponds to the schedule of the vessel. Vessels with the same number of containers to be loaded will, therefore, have the same EFT, independent of the yard density, and the number of container types.

4.7.2 Results for the mathematical model and enhancements

In this section, we report the computational results of the FSLP model and analyse the improvements achieved with the enhancements for this formulation. The results cover the lower bounds obtained in the root node of the branch-and-bound tree (x^{Root}), the best lower bound (x^{LP}), and the value of the obtained solution (x^{UB}). The $Gap(x^{UB})$ columns report the relative difference between the solution value and the best lower bound from the model, and lastly $t(s)$ is

[1] The value ∞ for κ means that the container swapping method is used in every iteration

the computation time in seconds. The first columns in Tables 4.3 and 4.4 describe the instance characteristics, with the number of containers ($|C|$), the number of container types ($|CT|$), the number of QCs ($|Q|$) and the density (D). The densities are Less Dense (LD), Scattered (S) and Uniform (U) as described in Section 4.7.1.

To evaluate the benefit of each of the enhancements, Tables 4.3 and 4.4 present the results for different versions of the model. The first one is the version with all of the enhancements added (FSLP+), after which results for the standard model with no enhancements (FSLP) are shown. For the rest of the models, enhancements are removed from the FSLP+ model, e.g. FSLP+ - (4.19) is the FSLP+ model with constraint (4.19) removed. Note that enhancements are removed one at a time. This is to show how badly the results are affected when an enhancement is not used in the formulation, and by that see if it is beneficial to add it.

There are many instances for which CPLEX cannot find any feasible solutions within the time limit (10800 s), this is indicated with '-'. Additionally '†' is used to symbolise that the execution was terminated due to lack of memory. In most of these cases no bound or solution could be computed before termination, but in four cases (FSLP+/60/25/2/S, FSLP+/60/10/2/U, FSLP+ - (4.21)/60/25/2/LD and FSLP+ - (4.21)/60/25/2/U) a solution had been found, in which case the time reported is the time at which the execution was terminated. To symbolise the best bound, or best solution found for the given instance among the alternative models, the corresponding value is written in **bold**.

Table 4.3 clearly shows that FSLP+ model outperforms the FSLP model. The FSLP+ model finds stronger bounds, and also better feasible solutions. Looking at the results for all models it can be seen that constraint (4.21) is the main contributor to the improvement of the bounds, but (4.17) & (4.18) also help to improve the bound. The performance of the three models FSLP+, FSLP+ - (4.19) and FSLP+ - (4.20) are all comparable to each other. The bounds all coincide with the best found, and they are all found in the root node and not improved hereafter. The value of the solutions are different, but for a given instance the solution values are close to each other for the three models. For what it's worth, FSLP+ - (4.20) seems to find the best solution most often.

For the 500 and 1000 instances, the model becomes intractable to solve; only a few number of feasible solutions are found within 3 hours.

Table 4.3: Performance of enhancements. '–' symbolises that no feasible solution were found within the time limit, and '†' is used for the instances where the execution was terminated due to insufficient memory.

				FSLP+					FSLP					FSLP+ - (4.17)&(4.18)				
$\|C\|$	$\|CT\|$	$\|Q\|$	D	x^{Root}	x^{LB}	x^{UB}	$Gap(x^{UB})$	$t(s)$	x^{Root}	x^{LB}	x^{UB}	$Gap(x^{UB})$	$t(s)$	x^{Root}	x^{LB}	x^{UB}	$Gap(x^{UB})$	$t(s)$
60	10	2	*LD*	**1500**	**1500**	1775	15.5%	10800	10	194	1735	88.8%	10800	1440	1440	**1710**	15.8%	10800
60	10	2	*S*	**1020**	**1020**	1030	1.0%	10800	30	93	1030	90.9%	10800	**1020**	**1020**	**1020**	0.0%	8246
60	10	2	*U*	**1640**	**1640**	1920	14.6%	9004†	41	139	2250	93.8%	10800	**1640**	**1640**	1875	12.5%	10800
60	25	2	*LD*	**2030**	**2030**	2345	13.4%	10800	0	336	2480	86.5%	10800	1790	1790	2285	21.7%	10800
60	25	2	*S*	**1360**	**1360**	1470	7.5%	6215†	0	257	1905	86.5%	10800	**1360**	**1360**	**1405**	3.2%	10800
60	25	2	*U*	**1490**	**1490**	1520	2.0%	10800	0	267	2225	88.0%	10800	**1490**	**1490**	**1510**	1.3%	10800
			Average				9.0%	9737				89.1%	10800				9.1%	10374
240	20	2	*LD*	**7850**	**7850**	14795	46.9%	10800	0	100	20170	99.5%	10800	6530	6530	-	-	10800
240	20	2	*S*	**4440**	**4440**	8225	46.0%	10800	0	50	109905	100.0%	10800	**4440**	**4440**	9040	50.9%	10800
240	20	2	*U*	**6720**	**6720**	11765	42.9%	10800	0	60	19765	99.7%	10800	**6720**	**6720**	11435	41.2%	10800
240	60	2	*LD*	**8230**	**8230**	11035	25.4%	10800	0	142	16565	99.1%	10800	6850	6850	**10650**	35.7%	10800
240	60	2	*S*	**5280**	**5280**	**6555**	19.5%	10800	0	79	11095	99.3%	10800	**5280**	**5280**	7310	27.8%	10800
240	60	2	*U*	**7250**	**7250**	10845	33.1%	10800	0	179	14005	98.7%	10800	**7250**	**7250**	10630	31.8%	10800
			Average				35.6%	10800				99.4%	10800				37.5%	10800
500	20	4	*LD*	**14390**	**14390**	-	-	10800	0	0	-	-	10800	12110	12110	-	-	10800
500	20	4	*S*	**8250**	**8250**	-	-	10800	0	0	-	-	10800	**8250**	**8250**	-	-	10800
500	20	4	*U*	**14350**	**14350**	-	-	10800	0	0	-	-	10800	13090	13090	-	-	10800
500	60	4	*LD*	**14460**	**14460**	-	-	10800	0	200	-	-	10800	12930	12930	-	-	10800
500	60	4	*S*	**11840**	**11840**	-	-	10800	0	60	-	-	10800	**11840**	**11840**	-	-	10800
500	60	4	*U*	**15500**	**15500**	264170	94.1%	10800	0	233	**30690**	99.2%	10800	14240	14240	-	-	10800
500	100	4	*LD*	**15390**	**15390**	-	-	10800	0	288	40225	99.3%	10800	13860	13860	138790	90.0%	10800
500	100	4	*S*	**9490**	**9490**	-	-	10800	0	140	24140	99.4%	10800	**9490**	**9490**	16895	43.8%	10800
500	100	4	*U*	**16230**	**16230**	**22625**	28.3%	10800	0	240	36700	99.3%	10800	14880	14880	-	-	10800
			Average				61.2%	10800				99.3%	10800				66.9%	10800
1000	20	4	*LD*	†	†	†	†	†	†	†	†	†	†	†	†	†	†	†
1000	20	4	*S*	†	†	†	†	†	†	†	†	†	†	†	†	†	†	†
1000	20	4	*U*	†	†	†	†	†	†	†	†	†	†	†	†	†	†	†
1000	60	4	*LD*	**27070**	**27070**	-	-	10800	0	0	-	-	10800	24460	24460	-	-	10800
1000	60	4	*S*	**17950**	**17950**	-	-	10800	0	0	-	-	10800	**17950**	**17950**	-	-	10800
1000	60	4	*U*	**40230**	**40230**	-	-	10800	0	0	-	-	10800	34320	34320	-	-	10800
1000	100	4	*LD*	**28740**	**28740**	-	-	10800	0	0	-	-	10800	25830	25830	-	-	10800
1000	100	4	*S*	**17840**	**17840**	-	-	10800	0	0	-	-	10800	**17840**	**17840**	-	-	10800
1000	100	4	*U*	**32180**	**32180**	-	-	10800	50	50	-	-	10800	29090	29090	-	-	10800
			Average					10800					10800					10800
Average							27.9%	10564				95.5%	10800				28.9%	10705

Table 4.4: Performance of enhancements. '–' symbolises that no feasible solution were found within the time limit, and '†' is used for the instances where the execution was terminated due to insufficient memory.

				FSLP+ - (4.19)					FSLP+ - (4.20)					FSLP+ - (4.21)				
$\|C\|$	$\|CT\|$	$\|Q\|$	D	x^{Root}	x^{LB}	x^{UB}	$Gap(x^{UB})$	$t(s)$	x^{Root}	x^{LB}	x^{UB}	$Gap(x^{UB})$	$t(s)$	x^{Root}	x^{LB}	x^{UB}	$Gap(x^{UB})$	$t(s)$
60	10	2	*LD*	**1500**	**1500**	1800	16.7%	10800	**1500**	**1500**	1720	12.8%	10800	101	222	1750	87.3%	10800
60	10	2	*S*	**1020**	**1020**	1025	0.5%	10800	**1020**	**1020**	**1020**	0.0%	5612	0	105	1030	89.8%	10800
60	10	2	*U*	**1640**	**1640**	**1790**	8.4%	10800	**1640**	**1640**	1820	9.9%	10800	0	121	1955	93.8%	10800
60	25	2	*LD*	**2030**	**2030**	2350	13.6%	10800	**2030**	**2030**	**2275**	10.8%	10800	241	632	2535	75.1%	7615†
60	25	2	*S*	**1360**	**1360**	1440	5.6%	10800	**1360**	**1360**	1580	13.9%	10800	0	307	1760	82.6%	10800
60	25	2	*U*	**1490**	**1490**	1530	2.6%	10800	**1490**	**1490**	1515	1.7%	10800	0	401	1740	76.9%	6568†
			Average				7.9%	10800				8.2%	9935				84.3%	9564
240	20	2	*LD*	**7850**	**7850**	14795	46.9%	10800	**7850**	**7850**	**12700**	38.2%	10800	1320	1403	19600	92.8%	10800
240	20	2	*S*	**4440**	**4440**	8225	46.0%	10800	**4440**	**4440**	**6400**	30.6%	10800	0	30	10045	99.7%	10800
240	20	2	*U*	**6720**	**6720**	-	-	10800	**6720**	**6720**	**11355**	40.8%	10800	0	30	17710	99.8%	10800
240	60	2	*LD*	**8230**	**8230**	10730	23.3%	10800	**8230**	**8230**	11705	29.7%	10800	1380	1500	15675	90.4%	10800
240	60	2	*S*	**5280**	**5280**	6680	21.0%	10800	**5280**	**5280**	6630	20.4%	10800	11	73	8265	99.1%	10800
240	60	2	*U*	**7250**	**7250**	**10185**	28.8%	10800	**7250**	**7250**	13330	45.6%	10800	0	170	16735	99.0%	10800
			Average				33.2%	10800				34.2%	10800				96.8%	10800
500	20	4	*LD*	**14390**	**14390**	-	-	10800	**14390**	**14390**	-	-	10800	2280	2280	-	-	10800
500	20	4	*S*	**8250**	**8250**	-	-	10800	**8250**	**8250**	-	-	10800	0	0	-	-	10800
500	20	4	*U*	**14350**	**14350**	-	-	10800	**14350**	**14350**	-	-	10800	1260	1260	-	-	10800
500	60	4	*LD*	**14460**	**14460**	-	-	10800	**14460**	**14460**	-	-	10800	1530	1750	-	-	10800
500	60	4	*S*	**11840**	**11840**	-	-	10800	**11840**	**11840**	**340870**	96.5%	10800	0	50	-	-	10800
500	60	4	*U*	**15500**	**15500**	-	-	10800	**15500**	**15500**	-	-	10800	1260	1501	-	-	10800
500	100	4	*LD*	**15390**	**15390**	**21730**	29.2%	10800	**15390**	**15390**	22695	32.2%	10800	1530	1753	37720	95.4%	10800
500	100	4	*S*	**9490**	**9490**	**16220**	41.5%	10800	**9490**	**9490**	-	-	10800	0	140	233945	99.9%	10800
500	100	4	*U*	**16230**	**16230**	-	-	10800	**16230**	**16230**	-	-	10800	1350	1629	41490	96.1%	10800
			Average				35.3%	10800				64.4%	10800				97.1%	10800
1000	20	4	*LD*	†	†	†	†	†	†	†	†	†	†	†	†	†	†	†
1000	20	4	*S*	†	†	†	†	†	†	†	†	†	†	†	†	†	†	†
1000	20	4	*U*	†	†	†	†	†	†	†	†	†	†	†	†	†	†	†
1000	60	4	*LD*	-	-	-	-	10800	**27070**	**27070**	-	-	10800	2610	2610	-	-	10800
1000	60	4	*S*	**17950**	**17950**	-	-	10800	**17950**	**17950**	-	-	10800	0	0	-	-	10800
1000	60	4	*U*	**40230**	**40230**	-	-	10800	**40230**	**40230**	-	-	10800	5910	5910	-	-	10800
1000	100	4	*LD*	**28740**	**28740**	-	-	10800	**28740**	**28740**	-	-	10800	2910	2910	-	-	10800
1000	100	4	*S*	**17840**	**17840**	-	-	10800	**17840**	**17840**	-	-	10800	0	0	-	-	10800
1000	100	4	*U*	**32180**	**32180**	-	-	10800	**32180**	**32180**	-	-	10800	3090	3090	-	-	10800
			Average					10800					10800					10800
Average							21.8%	10800				27.4%	10608				91.9%	10525

4.7.3 Heuristic results

We evaluate the performance of GRASP heuristic, and compare it with the results from the formulation FSLP+ and the lower bound model (LB-FSLP) presented in Section 4.5. Table 4.5 reports the results for the FSLP+ formulation, the lower bounds from LB-FSLP and the GRASP results on the instances from the benchmark. In Table 4.5, x^R is the lower bound from the LB-FSLP model and $t(s)$ reports the computational time to obtain these lower bounds. For the FSLP+ model, x^{LB} and x^{UB} is the lower bound, and upper bound obtained. $Gap(x^{UB})$ is the gap, comparing the best feasible solution with the best found lower bound. For the heuristic, each instance is solved ten times, to account for the randomness. The best solution and average solution for the ten runs are reported. For the GRASP heuristic, the following are reported in Table 4.5; the best solution (x^b), average solution ($\bar{x}$), best and average gap with respect to LB-FSLP ($Gap(x^b)$, $Gap(\bar{x})$) and the run time in seconds ($\bar{t}$). The average solution which are better than the FSLP+ formulation is written in **bold**.

We first evaluate the lower bounds obtained with LB-FSLP model. Table 4.5 points out that the lower bounds obtained with the LB-FSLP model are no worse than the x^{LB} values from the FSLP+ model, in some cases the LB-FSLP model even finds a better lower bound. Moreover, the bounds are computed in just 3 seconds on average. This indicates that the lower bounds obtained with LB-FSLP can be used to evaluate the performance of the GRASP heuristic.

The results in Table 4.5 show the GRASP heuristic finds feasible solutions for all of the instances, with an average gap of 10.9% in approx. 10 minutes on average. The gap calculates the relative difference to the lower bound, as seen in Section 4.7.2 only one instance has been solved to optimality, for the rest of the instances we have no indication of the quality of this lower bound, and how far it is from being optimal. The quality of the solutions found by the GRASP heuristic is stable with respect to the number of containers.

Looking at the 500 and 1000 container instances we see a clear tendency; the *Less Dense* instances require considerably more effort to solve, compared with the *Scattered* instances. The effort needed for the *Uniform* instances lies in between the two others. By looking at the underlying data, the explanation can be found in the time spent on the Improvement Phase. Excluding that time, there are only small deviations in the total time used.

Figure 4.4 shows how much the two improvement methods contribute to the quality of the final solution. The plots show how the average gap converges over time when using both improvement methods, only the container swapping method, only the vehicle reassignment method and no improvement methods. For each setting, the data is grounded on ten runs of each instance. The plots show that of the two, the container swapping method improves the solution the most, but is also the most time consuming one. From the plot, it is also clear that both of the improvement methods improves the solution noticeably.

To visualise the impact of the STIs, Figure 4.5 shows the probabilities for QC 2 during a single execution of the algorithm on the instance $60/25/2/LD$. The figure shows the probability of selecting a specific STI for the QC in an iteration. The probabilities are shown for 3 of the STIs, for the remaining STIs only a few improvements are found, and thus they become less and less probable. The probabilities for the STIs not shown in Figure 4.5 are below 0.1% from iteration 5000 until the end. The legends describe the range of the STI, i.e. the interval in which ν_q is

Table 4.5: Results overview: Comparison between the formulations and the GRASP heuristic. '–' is used when no feasible solution were found within the timelimit

				LB-FSLP		FSLP+			GRASP				
$\|C\|$	$\|CT\|$	$\|Q\|$	D	x^R	$t(s)$	x^{LB}	x^{UB}	$Gap(x^{UB})$	x^b	$\bar{x}$	$Gap(x^b)$	$Gap(\bar{x})$	$\bar{t}$ (s)
60	10	2	*LD*	1530	0.1	1500	1775	15.49%	1805	1814.5	15.24%	15.68%	9.9
60	10	2	*S*	1020	0.0	1020	1030	0.97%	1060	1067	3.77%	4.40%	7.5
60	10	2	*U*	1670	0.1	1640	1920	14.58%	1895	**1902.5**	11.87%	12.22%	8.8
60	25	2	*LD*	2060	0.0	2030	2345	13.43%	2435	2435	15.40%	15.40%	10.4
60	25	2	*S*	1360	0.0	1360	1470	7.48%	1425	**1436.5**	4.56%	5.32%	10.1
60	25	2	*U*	1490	0.0	1490	1520	1.97%	1545	1552.5	3.56%	4.02%	8.5
			Average		**0.1**						**9.07%**	**9.51%**	**9.2**
240	20	2	*LD*	7880	0.3	7850	14795	46.94%	9430	**9448**	16.44%	16.60%	80.5
240	20	2	*S*	4440	0.1	4440	8225	46.02%	4790	**4807.5**	7.31%	7.64%	36.2
240	20	2	*U*	6720	0.7	6720	11765	42.88%	8140	**8333.5**	17.44%	19.35%	36.8
240	60	2	*LD*	8260	0.1	8230	11035	25.42%	10105	**10123**	18.26%	18.40%	131.1
240	60	2	*S*	5280	0.1	5280	6555	19.45%	5660	**5706.5**	6.71%	7.47%	53.4
240	60	2	*U*	7580	0.2	7250	10845	33.15%	9065	**9140.5**	16.38%	17.07%	52.8
			Average		**0.2**						**13.76%**	**14.42%**	**65.1**
500	20	4	*LD*	14420	1.4	14390	-	-	15585	**15639**	7.48%	7.79%	538.7
500	20	4	*S*	8250	1.3	8250	-	-	9020	**9129.5**	8.54%	9.63%	75.0
500	20	4	*U*	14380	2.0	14350	-	-	15585	**15594.5**	7.73%	7.79%	381.9
500	60	4	*LD*	14520	4.2	14460	-	-	16130	**16225.5**	9.98%	10.51%	648.3
500	60	4	*S*	11840	0.2	11840	-	-	13005	**13080**	8.96%	9.48%	72.2
500	60	4	*U*	15530	0.2	15500	264170	94.13%	17125	**17173**	9.31%	9.57%	287.1
500	100	4	*LD*	15450	0.5	15390	-	-	17475	**17562**	11.59%	12.03%	627.7
500	100	4	*S*	9490	0.1	9490	-	-	10425	**10547.5**	8.97%	10.02%	73.6
500	100	4	*U*	16290	0.6	16230	22625	28.27%	18495	**18544**	11.92%	12.15%	282.8
			Average		**1.2**						**9.39%**	**9.89%**	**331.9**
1000	20	4	*LD*	21990	9.4	-	-	-	24225	**24277**	9.23%	9.42%	1984.0
1000	20	4	*S*	14570	1.7	-	-	-	16230	**16414**	10.23%	11.23%	292.2
1000	20	4	*U*	25980	49.9	-	-	-	28295	**28354**.5	8.18%	8.37%	1730.9
1000	60	4	*LD*	27100	3.2	27070	-	-	30000	**30074**	9.67%	9.89%	2821.9
1000	60	4	*S*	17950	0.7	17950	-	-	20305	**20599**	11.60%	12.85%	174.8
1000	60	4	*U*	40260	2.1	40230	-	-	44055	**44095.5**	8.61%	8.70%	1209.3
1000	100	4	*LD*	28770	1.7	28740	-	-	32330	**32372**	11.01%	11.13%	3129.7
1000	100	4	*S*	17840	0.5	17840	-	-	20150	**20347**	11.46%	12.32%	177.2
1000	100	4	*U*	32210	1.4	32180	-	-	36160	**36228**	10.92%	11.09%	2194.8
			Average		**7.8**						**10.10%**	**10.56%**	**1523.9**
Average					**2.8**						**10.41%**	**10.92%**	**571.6**

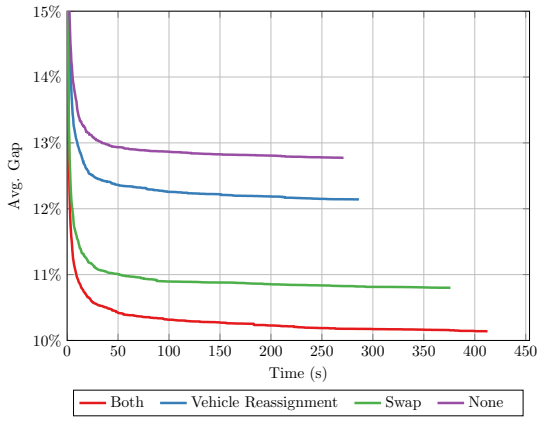


Figure 4.4: The impact of the improvement methods on time, and solution quality

randomly sampled when the STI is selected. Figure 4.5 shows that the most probable STIs (for this QC/instance combination) are towards the end of the full range for ν_q. The interval range for the high probability STIs is close to each other. This is as expected; if one STI provides good results, the following, or preceding is likely to perform semi good as well.

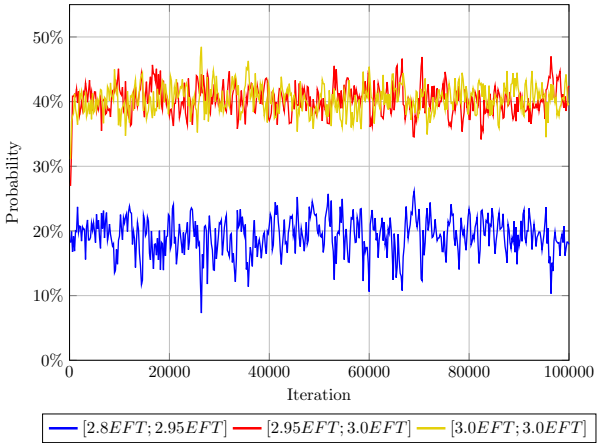
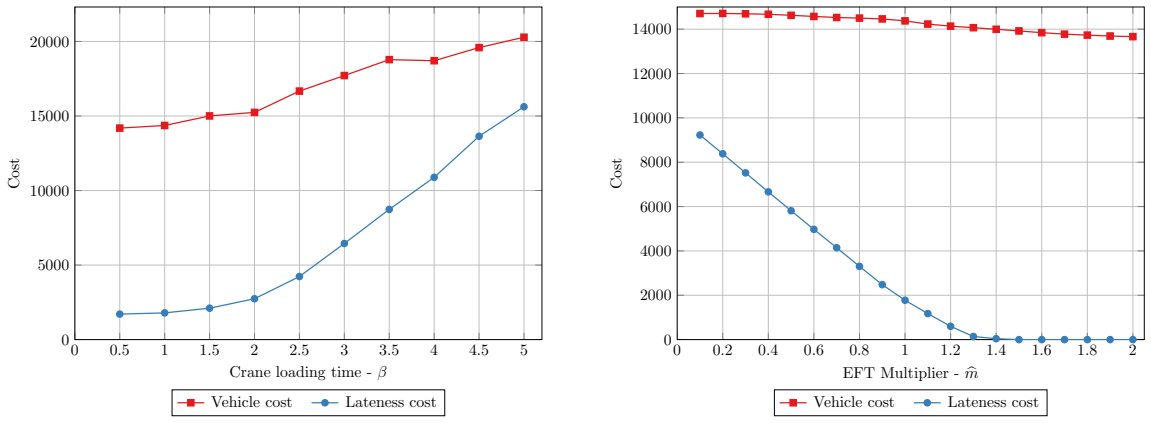


Figure 4.5: The STI probability during an execution of the algorithm on the instance $60/25/2/LD$ *for* QC 2

There are 2 parts to the objective function (4.1), first a cost of TVs services ($\alpha \sum_{s \in S}(End_s - Start_s)$) and a cost of lateness ($\gamma \Delta EFT$). Figure 4.6 shows the impact on the cost structure when changing the QC loading time, β and the EFT. Figure 4.6a analyses the impact of changing β and Figure 4.6b of changing the EFT. The data is an average of 10 runs of the heuristic on every instance. Figure 4.6a shows what we would expect, increasing β increases both the TV service cost and the cost of lateness. This is important for the terminal as crane operators will load the containers faster/slower depending on their skill level. Figure 4.6b shows the impact of changing the EFT. Here the EFT is recalculated as $\widehat{EFT} = \widehat{m}EFT$, where $\widehat{m}$ is a multiplier and EFT is the original EFT. Intuitively you would expect that a lower EFT means a higher cost of lateness, which Figure 4.6b confirms. Increasing the EFT mostly have an impact on the TV service cost for an instance when the lateness cost is 0 for that instance. This makes sense as fewer TVs can be used, and the loading be completed as expected. Using fewer TVs means less unproductive waiting time incurred by the loading order and β.



(a) Impact of β on the cost *(b) Impact of the EFT on the cost*

Figure 4.6: Impact of β and the EFT on the cost structure.

4.7.4 Hierarchical vs integrated planning: value of integration

We also investigate the cost savings with the integration of the operative stowage planning and TV assignment/scheduling. We compare results of the integrated FSLP with the hierarchical planning method. The hierarchical planning is simulated in two stages. In the first stage, we solve the lower bound model (LB-FSLP in Section 4.5) where the assignment decisions are solely made without considering the feasible TV scheduling.

The assignment decisions made in (4.22)-(4.32) will always generate feasible solutions for the integrated FSLP. This is because the remainder problem is a TV scheduling problem with predetermined assignments (x^s_{ip}). In the second stage, we fix the assignment variables obtained in the first stage and solve the TV scheduling problem. With the assignment fixed, the TV scheduling problem can be solved as a Linear Program. Define $t_p \in \mathbb{R}^+$ as the time the container for position p is dropped in front of the QC. Let c_p be the container to be loaded in position $p \in P$ (which is known). Let f_s and l_s be the first and last position served by TV $s \in S$ and let $\mathrm{A}(s)$ describe the full container/position assignment for that TV. With this, the TV scheduling problem can be modelled as follows:

$$\texttt{Min Z} = \alpha \sum_{s \in S} t_{l_s} - (t_{f_s} - 2\tau_{c_{f_s},f_s}) + \gamma \Delta EFT \tag{4.36}$$

subject to

$$t_p \geq t_{p\prime} + \beta \qquad \forall q \in Q, p \in P_q \setminus \{1\}, p\prime \prec\prec p \tag{4.37}$$

$$t_p \geq 2\tau_{c(p),p} \qquad \forall s \in S, p = f_s \tag{4.38}$$

$$t_p \geq t_{Prev(s,p)} + 2\tau_{c_p,p} \qquad \forall s \in S, p \in \mathrm{A}(s) \setminus \{f_s\} \tag{4.39}$$

$$z \geq t_p + \beta \qquad \forall q \in Q, p = |P_q| \tag{4.40}$$

$$\Delta EFT \geq z - EFT \tag{4.41}$$

The objective function (4.36) can be read in the same way as for the FSLP model. Here t_{l_s} is equivalent to End_s, and $(t_{f_s} - 2\tau_{c_{f_s},f_s})$ is equivalent to $Start_s$. In constraint (4.37), $p\prime$ is the

position loaded just before position p, and the constraint thus ensures that the loading time is respected. Constraints (4.38) and (4.39) ensure the transportation times are respected. For all vehicles, constraint (4.38) makes sure that the transportation time to the first assignment is respected. For all other assignments (4.39) have effect. Constraint (4.39) ensures that the earliest a TV can drop a container in front of the QC is the time it dropped its last container ($t_{Prev(s,p)}$) plus the time it takes to get the current container. Constraint (4.40)-(4.41) is equivalent to constraints (4.11) and (4.13) of the FSLP model.

Table 4.6: Integrated vs. hierarchical planning of FSLP

				Hierarchical				Integrated		Value of Integration	
$\|C\|$	$\|CT\|$	$\|Q\|$	D	x^{UB}	$t_1(s)$	$t_2(s)$	$t(s)$	x^{UB}	$\bar{t}$	Δ	%
60	10	2	*LD*	3585	0.1	0.0	0.1	1814.5	9.9	1770.5	49.39%
60	10	2	*S*	2010	0.0	0.0	0.0	1067.0	7.5	943.0	46.92%
60	10	2	*U*	4105	0.1	0.0	0.1	1902.5	8.8	2202.5	53.65%
60	25	2	*LD*	3895	0.0	0.0	0.0	2435.0	10.4	1460.0	37.48%
60	25	2	*S*	3530	0.0	0.0	0.0	1436.5	10.1	2093.5	59.31%
60	25	2	*U*	3760	0.0	0.0	0.0	1552.5	8.5	2207.5	58.71%
		Average									**50.91%**
240	20	2	*LD*	20880	0.3	0.0	0.3	9448.0	80.5	11432.0	54.75%
240	20	2	*S*	10975	0.1	0.0	0.1	4807.5	36.2	6167.5	56.20%
240	20	2	*U*	17455	0.7	0.0	0.7	8333.5	36.8	9121.5	52.26%
240	60	2	*LD*	19530	0.1	0.0	0.1	10123.0	131.1	9407.0	48.17%
240	60	2	*S*	15245	0.1	0.0	0.1	5706.5	53.4	9538.5	62.57%
240	60	2	*U*	20215	0.2	0.0	0.2	9140.5	52.8	11074.5	54.78%
		Average									**54.79%**
500	20	4	*LD*	33755	1.4	0.0	1.4	15639.0	538.7	18116.0	53.67%
500	20	4	*S*	19320	1.3	0.0	1.3	9129.5	75.0	10190.5	52.75%
500	20	4	*U*	32180	2.0	0.0	2.0	15594.5	381.9	16585.5	51.54%
500	60	4	*LD*	30235	4.2	0.0	4.2	16225.5	648.3	14009.5	46.34%
500	60	4	*S*	26185	0.2	0.0	0.2	13080.0	72.2	13105.0	50.05%
500	60	4	*U*	32395	0.2	0.0	0.2	17173.0	287.1	15222.0	46.99%
500	100	4	*LD*	32590	0.5	0.0	0.5	17562.0	627.7	15028.0	46.11%
500	100	4	*S*	21270	0.1	0.0	0.1	10547.5	73.6	10722.5	50.41%
500	100	4	*U*	36815	0.6	0.0	0.6	18544.0	282.8	18271.0	49.63%
		Average									**49.72%**
1000	20	4	*LD*	55815	9.4	0.0	9.4	24277.0	1984.0	31538.0	56.50%
1000	20	4	*S*	33375	1.7	0.0	1.7	16414.0	292.2	16961.0	50.82%
1000	20	4	*U*	58910	49.9	0.0	49.9	28354.5	1730.9	30555.5	51.87%
1000	60	4	*LD*	63880	3.2	0.0	3.2	30074.0	2821.9	33806.0	52.92%
1000	60	4	*S*	44300	0.7	0.0	0.7	20599.0	174.8	23701.0	53.50%
1000	60	4	*U*	82770	2.1	0.0	2.1	44095.5	1209.3	38674.5	46.73%
1000	100	4	*LD*	64735	1.7	0.0	1.7	32372.0	3129.7	32363.0	49.99%
1000	100	4	*S*	43030	0.5	0.0	0.5	20347.0	177.2	22683.0	52.71%
1000	100	4	*U*	69275	1.4	0.0	1.4	36228.0	2194.8	33047.0	47.70%
		Average									**51.42%**
Average											**51.48%**

In Table 4.6, instance properties are reported in the first four columns. The table is divided to sections presenting hierarchical, integrated and value of integration results. Column x^{UB} presents objective function values, while t variants present the time to obtain the x^{UB} values. In

hierarchical planning, $t(s)$ is the sum of two values, $t_1(s)$ and $t_2(s)$. The first value ($t_1(s)$) is the time to run first stage model (i.e. LB-FSLP), while the second value ($t_2(s)$) is the running time of the second stage model ((4.36) - (4.41)). Column Δ points out the cost reduction achieved (cost savings) by solving the integrated problem, while % is the percentage of decrease in the cost value (i.e. (Hierarchical-Integrated)/Hierarchical).

Results show that the average cost savings through integration are 51.48%. This suggests that there is a significant potential for savings for the terminal operators with such an integrated problem. Instances with 1000 containers obtain 51.42% savings, while 500, 240 and 60 containers instances result in 49.72%, 54.79% and 50.91% savings, respectively.

4.8 Conclusion and future research direction

In this paper, we have proposed a novel integrated container terminal problem. This problem focuses on the ship loading operations and aims at integrating the aspects of operative stowage planning with assignment and scheduling of transport vehicles. The problem has been formulated as a mathematical model. To improve the model, novel enhancements are described, and the computational results show that they improve the performance of the mathematical model. However, the exact method becomes computationally intractable for real-life instances. To deal with this, a GRASP heuristic has been implemented. The GRASP heuristic is shown to be scalable and can be used to find high-quality solutions in reasonable time.

The benefit of solving the integrated problem rather than in a hierarchical fashion has also been investigated. Results show that significant cost savings can be achieved with an efficient solution to the integrated problem.

We see many strong future research directions both on the problem definition and the solution method. With respect to the problem, we aim at going beyond some of the assumptions in the first place. The first clear addition would be integrating optimisation of the load sequencing within the FSLP. This extension will make the problem more complicated. However, the careful implementation of novel solution methods might obtain further cost savings. Another very promising research direction is to allow vehicle pooling and not restrict a given vehicle to only work for a single QC. Such an extension will allow for better utilisation of the TVs. Researchers could consider load and unload operations simultaneously, thus allowing for dual cycling to increase the utilisation of the QCs. Lastly, a terminal allocation problem with inter-terminal transshipment flows/movements, e.g. Zhen et al. (2016), can be integrated.

Regarding the solution method, we see three research perspectives. The first is to improve the exact solution method, either by decomposing the problem or reformulating the compact model. Secondly, we aim to improve the heuristic procedure, and lastly, we wish to devise a new method to calculate better lower bounds.

References

Alvarez, J. F. (2006). "A heuristic for vessel planning in a reach stacker terminal". In: *Journal of Maritime Research* 3.1, pp. 3–16.

Alvarez, J. F. (2008). "Optimization Algorithms for Maritime Terminal and Fleet Management". PhD thesis. Universitat Pompeu Fabra.

Ambrosino, D. and A. Sciomachen (2003). "Impact of yard organisation on the master bay planning problem". In: *Maritime Economics & Logistics* 5.3, pp. 285–300.

Ambrosino, D., A. Sciomachen, and E. Tanfani (2004). "Stowing a containership: the master bay plan problem". In: *Transportation Research Part A: Policy and Practice* 38.2, pp. 81–99. ISSN: 09658564.

Ansótegui, C. and K. Sellmann Meinolf and Tierney (2009). "A Gender-Based Genetic Algorithm for the Automatic Configuration of Algorithms". In: *Principles and Practice of Constraint Programming - CP 2009: 15th International Conference, CP 2009 Lisbon, Portugal, September 20-24, 2009 Proceedings*. Ed. by I. P. Gent. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 142–157.

Bian, Z., Q. Shao, and Z. Jin (2016). "Optimization on the container loading sequence based on hybrid dynamic programming". In: *Transport* 31.4, pp. 440–449.

Bierwirth, C. and F. Meisel (2015). "A follow-up survey of berth allocation and quay crane scheduling problems in container terminals". In: *European Journal of Operational Research* 244.3, pp. 675–689.

Bish, E., F. Chen, Y. Leong, B. Nelson, J. Ng, and D. Simchi-Levi (2005). "Dispatching vehicles in a mega container terminal". English. In: *OR Spectrum* 27.4, pp. 491–506. ISSN: 0171-6468.

Carlo, H. J., I. F. A. Vis, and K. J. Roodbergen (2013). "Seaside operations in container terminals: literature overview, trends, and research directions". In: *Flexible Services and Manufacturing Journal* 27.2, pp. 224–262.

Carlo, H. J., I. F. Vis, and K. J. Roodbergen (2014a). "Storage yard operations in container terminals: Literature overview, trends, and research directions". In: *European Journal of Operational Research* 235.2. Maritime Logistics, pp. 412–430.

Carlo, H. J., I. F. Vis, and K. J. Roodbergen (2014b). "Transport operations in container terminals: Literature overview, trends, research directions and classification scheme". In: *European Journal of Operational Research* 236.1, pp. 1–13.

Ding, Y., X.-J. Wei, Y. Yang, and T.-Y. Gu (2017). "Decision support based automatic container sequencing system using heuristic rules". In: *Cluster Computing* 20.1, pp. 239–252.

Feo, T. A. and M. G. C. Resende (1995). "Greedy Randomized Adaptive Search Procedures". In: *Journal of Global Optimization* 6.2, pp. 109–133.

Feo, T. A. and M. G. Resende (1989). "A probabilistic heuristic for a computationally difficult set covering problem". In: *Operations Research Letters* 8.2, pp. 67–71. ISSN: 0167-6377.

Goodchild, A. and C. Daganzo (2007). "Crane double cycling in container ports: Planning methods and evaluation". In: *Transportation Research Part B: Methodological* 41.8, pp. 875–891.

Imai, A., E. Nishimura, S. Papadimitriou, and K. Sasaki (2002). "The Containership Loading Problem". In: *International Journal of Maritime Economics* 4.2, pp. 126–148.

Imai, A., K. Sasaki, E. Nishimura, and S. Papadimitriou (2006). "Multi-objective simultaneous stowage and load planning for a container ship with container rehandle in yard stacks". In: *European Journal of Operational Research* 171.2, pp. 373–389.

Iris, Ç. and D. Pacino (2015). "A Survey on the Ship Loading Problem". English. In: *Computational Logistics*. Ed. by F. Corman, S. Voss, and R. R. Negenborn. Vol. 9335. Lecture Notes in Computer Science. Springer International Publishing, pp. 238–251.

Iris, Ç., D. Pacino, and S. Ropke (2017). "Improved Formulations and an Adaptive Large Neighborhood Search Heuristic for the Integrated Berth Allocation and Quay Crane Assignment Problem". In: *Transportation Research Part E: Logistics and Transportation Review* 105, pp. 123–147.

Iris, Ç., D. Pacino, S. Ropke, and A. Larsen (2015). "Integrated Berth Allocation and Quay Crane Assignment Problem: Set partitioning models and computational results". In: *Transportation Research Part E: Logistics and Transportation Review* 81, pp. 75–97.

Ji, M., W. Guo, H. Zhu, and Y. Yang (2015). "Optimization of loading sequence and rehandling strategy for multi-quay crane operations in container terminals". In: *Transportation Research Part E: Logistics and Transportation Review* 80.0, pp. 1–19. ISSN: 1366-5545.

Jiang, X. J. and J. G. Jin (2017). "A branch-and-price method for integrated yard crane deployment and container allocation in transshipment yards". In: *Transportation Research Part B: Methodological* 98, pp. 62–75.

Jung, S. and K. Kim (2006). "Load scheduling for multiple quay cranes in port container terminals". English. In: *Journal of Intelligent Manufacturing* 17.4, pp. 479–492. ISSN: 0956-5515.

Kim, K. H., J. S. Kang, and K. R. Ryu (2004). "A beam search algorithm for the load sequencing of outbound containers in port container terminals". In: *OR Spectrum* 26.1, pp. 93–116.

Kim, K. H. and K. Y. Kim (1999). "An Optimal Routing Algorithm for a Transfer Crane in Port Container Terminals". In: *Transportation Science* 33.1, pp. 17–33.

Kim, K. H. and H. Lee (2015). "Container Terminal Operation: Current Trends and Future Challenges". In: *Handbook of Ocean Container Transport Logistics: Making Global Supply Chains Effective*. Ed. by C.-Y. Lee and Q. Meng. Cham: Springer International Publishing, pp. 43–73. ISBN: 978-3-319-11891-8.

Kim, K. H. and Y.-M. Park (2004). "A crane scheduling method for port container terminals". In: *European Journal of Operational Research* 156.3, pp. 752–768.

Kontoravdis, G. and J. F. Bard (1995). "A GRASP for the Vehicle Routing Problem with Time Windows". In: *ORSA Journal on Computing* 7.1, pp. 10–23. DOI: 10.1287/ijoc.7.1.10.

Lee, Y. H., J. Kang, K. R. Ryu, and K. H. Kim (2005). "Optimization of container load sequencing by a hybrid of ant colony optimization and tabu search". In: *Advances in Natural Computation*. Ed. by Y. S. O. Lipo Wang Ke Chen. Springer, pp. 1259–1268.

Legato, P., R. Trunfio, and F. Meisel (2012). "Modeling and solving rich quay crane scheduling problems". In: *Computers & Operations Research* 39.9, pp. 2063–2078.

Li, C.-L. and G. L. Vairaktarakis (2004). "Loading and unloading operations in container terminals". In: *IIE Transactions* 36.4, pp. 287–297.

Meisel, F. and M. Wichmann (2010). "Container sequencing for quay cranes with internal reshuffles". In: *OR spectrum* 32.3, pp. 569–591.

Monaco, M. F., M. Sammarra, and G. Sorrentino (2014). "The Terminal-Oriented Ship Stowage Planning Problem". In: *European Journal of Operational Research* 239.1, pp. 256–265.

Pacino, D., A. Delgado, R. M. Jensen, and T. Bebbington (2011). "Fast Generation of Near-Optimal Plans for Eco-Efficient Stowage of Large Container Vessels". In: *Computational Logistics*. Ed. by J. W. Böse, H. Hu, C. Jahn, X. Shi, R. Stahlbock, and S. Voß. Vol. 6971. Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 286–301.

Pardalos, L. and M. Resende (1994). "A greedy randomized adaptive search procedure for the quadratic assignment problem". In: *Quadratic Assignment and Related Problems, DIMACS Series on Discrete Mathematics and Theoretical Computer Science* 16, pp. 237–261.

Parreno, F., D. Pacino, and R. Alvarez-Valdes (2016). "A GRASP algorithm for the container stowage slot planning problem". In: *Transportation Research Part E: Logistics and Transportation Review* 94, pp. 141–157. ISSN: 1366-5545.

Ropke, S. and D. Pisinger (2006). "An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows". In: *Transportation Science* 40.4, pp. 455–472.

Steenken, D., S. Voß, and R. Stahlbock (2004). "Container terminal operation and operations research-a classification and literature review". In: *OR Spectrum* 26.1, pp. 3–49.

Steenken, D., T. Winter, and U. T. Zimmermann (2001). "Stowage and transport optimization in ship planning". In: *Online optimization of large scale systems*. Ed. by J. R. Martin Groetschel Sven O. Krumke. Springer, pp. 731–745.

Turkogullari, Y. B., Z. C. Taskin, N. Aras, and I. K. Altinel (2016). "Optimal berth allocation, time-variant quay crane assignment and scheduling with crane setups in container terminals". In: *European Journal of Operational Research* 254.3, pp. 985–1001.

UNCTAD, S. (2015). "Review of Maritime Transport 2014". In: *United Nations Conference on Trade and Development*.

Zeng, Q. and Z. Yang (2009). "Integrating simulation and optimization to schedule loading operations in container terminals". In: *Computers & Operations Research* 36.6, pp. 1935–1944.

Zhen, L., S. Wang, and K. Wang (2016). "Terminal allocation problem in a transshipment hub considering bunker consumption". In: *Naval Research Logistics (NRL)* 63.7, pp. 529–548.

4.A Container swapping: Speed up

As mentioned in Section 4.6.2.4, calculating the improvement from performing a swap is the most computationally expensive part of the container swapping improvement method. This appendix describes how costs calculated previously can be reused, thus speeding up the overall method. Algorithm 5 describes how the improvement of swapping two containers (in terms of objective value) is calculated.

Algorithm 5 `EvaluateSwapImprovement`

Input: x, c_1, c_2

if $\tau_{c_1,p(c_1)} = \tau_{c_2,p(c_1)}$ **and** $\tau_{c_2,p(c_2)} = \tau_{c_1,p(c_2)}$ **then**
  if $q(c_1) = q(c_2)$ **then**
    `swapAndUpdateTimes`(x, c_1, c_2)
    workSaved $\leftarrow$ `calculateWorkSaved`(x, c_1, c_2)
    newFinishTime $\leftarrow \max\left(\texttt{calculateFinishTime}(x, c_1, c_2), \max_{q \in Q \setminus \{q(c_1)\}} (\texttt{craneFinishTime}(x, q))\right)$
    lateSaved $\leftarrow$ `calculateLateSaved`$(x,$ newFinishTime$)$
    return α·workSaved+γ·lateSaved
  else
    workSaved$_1$, newFinishTime$_1$ $\leftarrow$ `saveSwapProfit`$(x, p(c_1), \tau_{c_2,p(c_1)}, c_1, c_2)$
    workSaved$_2$, newFinishTime$_2$ $\leftarrow$ `saveSwapProfit`$(x, p(c_2), \tau_{c_1,p(c_2)}, c_2, c_1)$
    workSaved $\leftarrow$ workSaved$_1$ + workSaved$_2$
    newFinishTime $\leftarrow \max\left(\text{newFinishTime}_1, \text{newFinishTime}_2, \max_{q \in Q \setminus \{q(c_1), q(c_2)\}} (\texttt{craneFinishTime}(x, q))\right)$
    lateSaved $\leftarrow$ `calculateLateSaved`$(x,$ newFinishTime$)$
    return α·workSaved+γ·lateSaved
  end if
else
  return 0
end if

Consider two containers c_1 and c_2 their respective position in the current solution $p(c_1)$, $p(c_2)$, and their QCs $q(c_1)$ and $q(c_2)$. First observe that if the swap implies no change in the driving times, then the objective will not change due to the swap (lines 1, 16-17). If the driving times are different, the algorithm considers two cases; $q(c_1)$ and $q(c_2)$ are the same, or they are different (lines 2 and 8).

In the case when the two considered QCs are identical, the improvement is calculated by swapping the containers and iteratively updating the delivery times for the container-assignments (line 3). With the times updated the work saved and the time saved in lateness is easily computed by comparing with the solution x (lines 4-6).

In the case when the two considered QCs are different, the cost calculation is done container for container (lines 9-10). The procedure `saveSwapImprovement`(x, p, τ, c_1, c_2) takes two containers as input and returns the work saved and the new finishing time for QC $q(c_1)$ when swapping containers c_1 and c_2. Hereafter the cost improvement is calculated similarly to the previous case (lines 11-13).

The fundamental idea behind `saveSwapImprovement`(x, p, τ, c_1, c_2) is that you can compute the costs when needed, and reuse these costs when appropriate instead of calculating it again. Keep in mind that this swapping method does not put a position on a new TV, but simple changes the container to be loaded on a position to another compatible container. The important thing here is the driving time to the new container. Consider the following situation; you have two candidates swaps $\{c_1, c_2\}$ and $\{c_1, c_3\}$ where $q(c_1) \neq q(c_2)$ and $q(c_1) \neq q(c_3)$. The changes in the cost for QC $q(c_1)$ only depends on the driving time to the new container. If the driving time from position $p(c_1)$ to container c_2 is the same as the driving time to c_3, then you know the cost for QC $q(c_1)$ will be the same for these two swaps. Therefore you really only need to calculate the cost once, store it and reuse it for the second case. This is exactly what is done in `saveSwapImprovement`(x, p, τ, c_1, c_2) as seen in Algorithm 6.

Algorithm 6 `saveSwapImprovement`

Input: x, p, τ, c_1, c_2
1: workSaved $\leftarrow \mathbf{W}(c_1, \tau)$
2: newFinishTime $\leftarrow \mathbf{F}(c_1, \tau)$
3: **if** workSaved $= \emptyset$ **then**
4: `swapAndUpdateTimes`(x, c_1, c_2)
5: workSaved $\leftarrow$ `calculateWorkSaved`(x, c_1, c_2)
6: newFinishTime $\leftarrow$ `calculateFinishTime`(x, c_1, c_2)
7: $\mathbf{W}(c_1, \hat{\tau}) \leftarrow$ workSaved
8: $\mathbf{F}(c_1, \hat{\tau}) \leftarrow$ newFinishTime
9: **end if**
10: **return** workSaved, newFinishTime

The costs, as well as the finishing time for QC $q(c_1)$, are stored in $\mathbf{W}(c_1, \tau)$ and $\mathbf{F}(c_1, \tau)$. If we have not calculated the cost yet we calculate the cost and store it (lines 3-8), otherwise, we reuse what we have calculated (lines 1-2). $\mathbf{W}$ and $\mathbf{F}$ are stored globally and can be reused between moves within the improvement method. Only when making changes to a QC q do we have to reset the costs for all containers using QC q, as these costs are no longer ensured to be valid. When doing so the cost for picking up a container with the same driving time as the current one is set to 0, and the finish time is set to the finishing time for QC q

$$\mathbf{W}(c, \tau_{c,p(c)}) = 0 \qquad \forall c \in \{c' \in C \mid q(c') = q\}$$
$$\mathbf{F}(c, \tau_{c,p(c)}) = \texttt{craneFinishTime}(x, q) \qquad \forall c \in \{c' \in C \mid q(c') = q\}$$

Part II

Additional Work

Chapter 5

Additional work for the Cargo Mix Problem with Block Stowage

5.1 Introduction

This chapter extends the heuristic of the paper "A Matheuristic for the Cargo Mix Problem with Block Stowage" presented in Chapter 2. The extensions focus on Phase I and try to improve the block assignment. In the following, we will describe an iterative approach and a MIP model for Phase I.

While interesting from a methodological point of view, the results from the Iterative method does not justify the increase in the runtime. The MIP model for Phase I approach improves the results, and only uses little extra time. Results for both of the methods show there is still room for improvement.

The rest of the chapter is organised as follows; Section 5.2 describes the iterative approach, and Section 5.3 describe the Phase I model. In Section 5.4 the results for the two methods are reported, and lastly Section 5.5 concludes on the work presented in this chapter.

5.2 Iterative method

To improve the solutions found by the CMPBSHLP heuristic described in Section 2.5, an iterative method has been developed. The iterative method is an improvement heuristic that improves a current solution by defining a neighbourhood that is explored using a MIP.

First define a schedule, s, as a block assignment for a full rotation. Let this schedule be described

by the parameter b^d_{sp} defined as follows.

$$b^d_{sp} = \begin{cases} 1 & \text{If schedule } s \in \Omega \text{ assigns destination } d \in \mathcal{P} \text{ as the discharge port when in port } p \in \mathcal{P} \\ 0 & \text{Otherwise.} \end{cases}$$

Where Ω is the full set of possible schedules. Furthermore, define the variable r_{bs}

$$r_{bs} = \begin{cases} 1 & \text{If block } b \in \mathcal{B} \text{ uses schedule } s \in \Omega \\ 0 & \text{Otherwise.} \end{cases}$$

With this constraint (2.9), (2.10) and (2.14) can be exchanged with the following

$$\sum_{s \in \Omega} r_{bs} \leq 1 \qquad \forall b \in \mathcal{B} \tag{5.1}$$

$$y^{tc}_b \leq a^{tc} \sum_{s \in \Omega} b^{d^t}_{sp} r_{bs} \qquad \forall b \in \mathcal{B}, c \in \mathcal{C}, p \in \mathcal{P}, t \in \mathcal{T}^{On}_p \tag{5.2}$$

$$r_{bs} \in \{0, 1\} \qquad \forall b \in \mathcal{B}, s \in \Omega \tag{5.3}$$

Here constraint (5.1) substitutes for constraint (2.9) ensuring that at most a single schedule is chosen for a block, and (5.2) is similar to (2.10) and ensures we can only place a container if the container matches the discharge port as determined by the schedule. Now consider the following model

`Max` (2.1)

`Subject to:`

$$(2.2) - (2.8) \tag{5.4}$$

$$(5.1) - (5.3) \tag{5.5}$$

$$(2.11) - (2.13) \tag{5.6}$$

This model is another way to model the Cargo Mix Problem with Block Stowage, and can thus be used as a substitute for the model presented in Section 2.4. The difference between the two models is that here, the block assignment is defined on schedule variables instead of the σ^d_{bp} variables.

The size of Ω grows exponentially with the number of ports in the rotation, and it is inefficient to enumerate all these for all but the smallest instances. Instead, let Ω_{gen} be the set of unique schedules that are generated in Phase I as described in Section 2.5.1. The iterative method consists of 2 main steps, namely an Improvement step where a set of new block assignments are found, and an Evaluation step where these are evaluated to select the best.

5.2.1 Improvement Step

To get a feasible solution to improve, the CMPBSHLP heuristic from Section 2.5 is used, and $\widehat{r}_{bs}$ is set to 1 if schedule $s \in \Omega_{gen}$ is used for block b, and 0 otherwise. Additionally, let $\widehat{y}^{tc}_b$ be

the solution obtained from Phase II. Define the set $\mathcal{BA}$ as the set of $\langle b, s\rangle$ combinations where $\widehat{r}_{bs}$ is 1 in the current solution, i.e $\mathcal{BA} = \{b \in \mathcal{B}, s \in \Omega_{gen} : \widehat{r}_{bs} = 1\}$. The set $\mathcal{BA}$ thus defines the block assignment. To improve the solution, we would like to find a new and improved block assignment. Instead of trying random block assignments and evaluate them, we will formulate a MIP model to find a new block assignment that is similar to the current one. We will thus define a neighbourhood using the following constraint.

$$\sum_{\langle b,s\rangle \notin \mathcal{BA}} r_{bs} + \sum_{\langle b,s\rangle \in \mathcal{BA}} 1 - r_{bs} \leq k \tag{5.7}$$

Where k is a parameter defining the size of the neighbourhood. With this constraint at most $\lfloor \frac{k}{2} \rfloor$ blocks can change schedule to another schedule from the set Ω_{gen}. The full model solved in the improvement step is thus

`Max` (2.1)

`Subject to:`

(2.2)
(2.4) – (2.8)
(5.1) – (5.3)
(5.7)
(2.11) – (2.12)
(2.17)
(2.19)

When solving this model the solution defined by $\widehat{y}_b^{tc}$ is used as a warm-start. In the improvement step we are interested in finding a new block assignment, thus we do not need to solve to optimality. Instead, only the root node is solved to benefit from the cuts applied by CPLEX. If after this, no unexplored block assignment is found we continue solving until a new block assignment is found. Let $\mathcal{IS}$ be the set of all new block assignments found. From the improvement step, we go to an evaluation step, where each block assignment $\mathcal{BA} \in \mathcal{IS}$ is evaluated, and the best one is used in the next iteration of the improvement step.

5.2.2 Evaluation Step

All newly found block assignments are evaluated to select the one that seems the most promising. For all block assignments $\mathcal{BA} \in \mathcal{IS}$ define $\widehat{\sigma}_{bp}^{d}(\mathcal{BA})$ as follows

$$\widehat{\sigma}_{bp}^{d}(\mathcal{BA}) = \begin{cases} 1 & \exists s \in \Omega_{gen} : \langle b, s\rangle \in \mathcal{BA} \wedge b_{sp}^{d} \widehat{r}_{bs} = 1 \\ 0 & \text{Otherwise.} \end{cases} \qquad \forall \mathcal{BA} \in \mathcal{IS}, b \in \mathcal{B}, p \in \mathcal{P}, d \in \mathcal{P}$$

The interpretation of $\widehat{\sigma}_{bp}^{d}(\mathcal{BA})$ is thus similar to $\widehat{\sigma}_{bp}^{d}$ from Section 2.5.2, as it defines the block assignment for $\mathcal{BA}$. Now let Phase II($\mathcal{BA}$) be the model solved in Phase II of the CMPBSHLP

heuristic (see Section 2.5.2) with $\widehat{\sigma}^d_{bp}(\mathcal{BA})$ as the block assignment. To evaluate the block assignment $\mathcal{BA} \in \mathcal{IS}$ we solve the root node of the model Phase II($\mathcal{BA}$). Stopping after the root node yields a good upper bound, $x^{ub}(\mathcal{BA})$, which can be used to evaluate the quality of the best possible solution found with the block assignment $\mathcal{BA}$. The upper bound is compared to the upper bound for the current block assignment. If we in the evaluation step find a new bound improving block assignment, $\mathcal{BA}$, we again solve Phase II($\mathcal{BA}$), but here we continue solving until the optimality tolerance η is reached. This is done to improve the warm start used in the improvement step. Hereafter we go back to the improvement step and define a new neighbourhood and start looking for a new set of block assignments. If we in the evaluation step fail to find a bound improving schedule we do not change the block assignment and instead continue solving the improvement model with the saved Branch and bound tree.

The termination criteria for this algorithm can either be based on convergence, a time limit, or the number of iterations. Our termination criteria is based on the runtime.

5.2.3 Increasing the Neighborhood

If we fail to find a new block assignment and instead solve the improvement step model to optimality, then we know we have a locally optimal solution. In this case, we increase the size of the neighbourhood and solve again. To control the neighbourhood size we define 4 parameters, k^{init}, k^+, k^{max} and ω^+. Here k^{init} is the initial value of k, k^+ describes the number to add to the neighbourhood size k, and k^{max} is the maximum allowed value of k. If the maximum value of k is reached and we solve the improvement model to optimality, we will find and add ω^+ new schedules to the set Ω_{gen}. When generating new schedules, we use the same procedure as described in Section 2.5.1, with average TEU capacity and average reefer TEU capacity instead of block capacities. After solving the described longest path problem, we check if this schedule already has been generated, if not we add it to the set Ω_{gen}, and check if we have added ω^+ new schedules, if not we continue generating schedules.

The Iterative Method as described here does not ensure feasibility. The weight constraints (2.3) is relaxed, and the container assignment variables are continuous. We could, however, keep all block assignments in memory as well as the values $\widehat{y}^{bc}_t(\mathcal{BA})$, and solve multiple Phase III models (see Section 2.5.3) for these inputs and then return the best feasible solution.

5.3 Phase I model

In Phase I of the CMPBSHLP heuristic, schedules are generated by solving a longest path problem. However, the sequence in which the blocks are treated is not considered. Changing the sequence will result in another block assignment, one that possibly is better. However, there is no way to determine a priori if a specific sequence will result in a better or worse solution, and trying all sequences is computationally intractable.

In this section, we will try to alleviate this by formulating and solving, a mixed integer program-

ming model that, based on the uniquely generated schedules, assign schedules to blocks. The graph-based method described in Section 2.5.1 is executed as normally, but we also keep track of the number of TEUs of transport t assigned to a schedule s. After the graph-based method is finished, we can define Ω_{gen} as the set of uniquely generated schedules, and a_s^t as the number of TEUs of transport $t \in \mathcal{T}$ assigned schedule $s \in \Omega_{gen}$. In the model we assign schedules to blocks, ensuring the blocks TEU capacity is satisfied. To keep the model simple we only consider the TEU capacity. To further lessen the complexity we also enforce an upper bound corresponding to the number of TEUs assigned to a schedule in the graph-based method.

The decision variables in the model are y_{bs} and x_{bs}^t. Where y_{bs} is a binary variable denoting whether or not block $b \in \mathcal{B}$ is assigned schedule $s \in \Omega_{gen}$, and x_{bs}^t is the number of TEUs of transport $t \in \mathcal{T}$ assigned block $b \in \mathcal{B}$ from schedule $s \in \Omega_{gen}$. Below the sets, variables and parameters are summarized.

Sets:

Ω_{gen}	Set of unique generated schedules
$\mathcal{B}$	Set of blocks
$\mathcal{P}$	Set of ports
$\mathcal{T}$	Set of transports
$\mathcal{T}_p^{ON}$	Set of transports that visits port $p \in \mathcal{P}$

Variables:

$y_{bs} \in \{0,1\}$	1 if block $b \in \mathcal{B}$ is assigned schedule $s \in \Omega_{gen}$, 0 otherwise.
$x_{bs}^t \in \mathbb{N}$	Number of TEUs of transport $t \in \mathcal{T}$ assigned block $b \in \mathcal{B}$ from schedule $s \in \Omega_{gen}$

Parameters:

a_s^t	Number of TEUs of transport $t \in \mathcal{T}$ assigned schedule $s \in \Omega_{gen}$.
k_b	TEU capacity for block $b \in \mathcal{B}$
p^t	Number of ports visited by transport $t \in \mathcal{T}$

With this, the problem can be formulated as follows.

$$\texttt{Max Z} = \sum_{b \in \mathcal{B}} \sum_{s \in \Omega_{gen}} \sum_{t \in \mathcal{T}} p^t x_{bs}^t \tag{5.8}$$

`Subject to:`

$$\sum_{s \in \Omega_{gen}} y_b^s = 1 \qquad \forall b \in \mathcal{B} \tag{5.9}$$

$$\sum_{s \in \Omega_{gen}} \sum_{t \in \mathcal{T}_p^{ON}} x_{bs}^t \leq k_b \qquad \forall b \in \mathcal{B}, p \in \mathcal{P} \tag{5.10}$$

$$x_{bs}^t \leq a_s^t y_{bs} \qquad \forall b \in \mathcal{B}, s \in \Omega_{gen}, t \in \mathcal{T} \tag{5.11}$$

$$\sum_{b \in \mathcal{B}} x_{bs}^t \leq a_s^t \qquad \forall s \in \Omega_{gen}, t \in \mathcal{T} \tag{5.12}$$

$$y_{bs} \in \{0,1\} \qquad \forall b \in \mathcal{B}, s \in \Omega_{gen} \tag{5.13}$$

$$x_{bs}^t \in \mathbb{N} \qquad \forall b \in \mathcal{B}, s \in \Omega_{gen}, t \in \mathcal{T} \tag{5.14}$$

Here the objective (5.8) aims to optimise the intake. This is the most obvious objective when the overall objective is to maximise the intake, but the same objective is used when the overall revenue is optimised. If we instead were to optimise the revenue in the model (5.8)-(5.14) we would need to distinguish between the different container types, adding extra complexity to the model. We do not believe it is worth it to add the extra complexity and we believe optimising the intake here will also result in a good block assignment when the objective is changed in Phase II and Phase III. Constraint (5.9) ensures exactly one schedule is assigned every block, and constraint (5.10) enforces that the block TEU capacity is satisfied. Constraint (5.11) makes sure that we can only assign container to a schedule if the block uses the corresponding schedule, and in constraint (5.12) it is ensured we do not assign more containers to a schedule than we did in the graph-based method. Lastly, constraint (5.13) and (5.14) defines the variable domains.

We will call this revised method for R-CMPBSHLP, which uses Phase I as described here. Phase II and Phase III remains unchanged, and is as described in Section 2.5.

5.4 Results

5.4.1 Results for the Iterative method

When testing the iterative method the following parameters were used; $k^{init} = 4$, $k^{+} = 2$, $k^{max} = 10$, and $\omega^{+} = 1$. The algorithm was given 30 minutes to optimise and has been tested when optimising intake. In Table 5.1 the result from the iterative method is compared with the result obtained after Phase II of the CMPBSHLP heuristic.[1] In Table 5.1 the results for the CMPBSHLP heuristic is the same as the result reported for Phase II in Table 2.5. Table 5.1 shows the results for the smallest of the instances.[2] In the table, $\bar{x}$ is the average value of the solution, $\bar{t}$ is the average time, and for the iterative method #Ite is the average number of iterations completed within the time limit.

The iterative method improves the solutions found by the heuristic, but not enough to justify the steep increase in the runtime needed. The improvement was to be expected as the CMPBSHLP heuristic is used to define the start solution and thus cannot perform any worse than the heuristic. The major benefits for the original heuristic are the scalability and the fast response time. As shown in the Chapter 2 it scales well with the number of ports and manages to keep running

[1] As mentioned, the iterative method does not ensure a feasible solution, and thus it is more fair to compare it with the results obtained after Phase II in the heuristic.

[2] Due to an out of memory error only 11 out of the remaining 90 instances completed the run.

Table 5.1: Results for the Iterative Method

	UB	CMPBSHLP		Iterative		
$\|P\|$	$\bar{x}$	$\bar{x}$	$\bar{t}$	#Ite	$\bar{x}$	$\bar{t}$
4	53265	50985	1.1	15.5	52899	1800
5	68153	63883	1.8	10.7	65943	1800
6	85935	82100	1.6	6	83986	1800
7	102409	94995	3.4	3.4	98193	1800
8	122125	113379	4.5	1.6	113959	1800
9	131360	122824	3.7	1.3	123869	1800
10	149285	139182	6.7	2.1	139335	1800
11	170912	156014	6.7	1	157050	1800

times low. The iterative method does not scale well as can be seen in the number of iterations performed. When the number of ports is above 7 barely any iterations are performed, which means that the root node in the model solved in the improvement step cannot be solved, even with the small neighbourhood ($k^{init} = 4$).

5.4.2 Results for the Phase I Model

The revised heuristic is tested on same instances as in Chapter 2 and a time limit of 60 seconds is used for the Phase I Model. Table 5.2 presents the results when optimizing the intake. Here the results are compared with the result from the heuristic presented in Chapter 2. In the table $\bar{x}$ is the average solution, $\bar{t}$ is the time in seconds, and $\bar{t}_I$ is the time for the model used in Phase I.

Table 5.2: Results for the R-CMPBSHLP Method

	UB	CMPBSHLP		R-CMPBSHLP	
$\|P\|$	$\bar{x}$	$\bar{x}$	$\bar{t}$	$\bar{t}_I$	$\bar{x}$
4	53265	50351	1.3	0.4	50615
5	68153	62917	2.1	0.9	63713
6	85935	81209	2	2.3	81470
7	102409	93191	3.9	2.8	95200
8	122125	112787	5.2	3.9	112988
9	131360	122022	4.4	6.7	122661
10	149285	138485	7.5	8.1	139370
11	170912	155353	7.4	0.7	155561
12	185836	168760	17.4	8.3	169500
13	201501	185299	17.4	10.4	185932
14	212244	196018	13.2	4.0	197135
15	231770	212763	18.9	4.2	213150
16	249557	229117	23.2	9.7	229541
17	265357	245174	19.4	8.4	245197
18	280448	258001	43.2	13.5	257741
19	296581	273704	25.7	14.2	275290
20	308392	285586	24.6	29.8	286612
Average	183243	168867	13.9	7.5	169510

The result in Table 5.2 shows that the model described in Section 5.3 can be solved using

7.5 seconds on average. Of the 170 instances, only 7 is terminated due to the time limit (60 seconds). For these 7 instances, the average gap at termination was 0.70%. This illustrates that the model can be solved relative fast. However, looking at the solution quality, we only see minor improvements. This is both good and bad; good because this means that the graph-based method does a good job at assigning schedules to blocks. However, bad because, this means that the revised method does not yield significantly better solutions.

5.5 Conclusion

In this chapter, we have proposed two methods for improving the block assignment for the Cargo Mix Problem with Block Stowage. First, an Iterative method is described, and then an extension to the heuristic presented in Chapter 2 is proposed.

An iterative method must focus on changing the block assignment, and to evaluate a block assignment we must solve a MIP. Solving the MIP is time-consuming, and therefore we do not believe randomly assigning schedules is a good idea as we will spend most of the time evaluating the block assignments, incurring a high run time. Here we have tried to define a small neighbourhood which is then explored by a MIP, in that way we can evaluate the block assignments while searching for a new. However, the method does not scale well, and the results cannot justify the increase in the runtime. Therefore, for an iterative approach to work, we believe that much more work or a radically different approach is needed.

The proposed Phase I model extension aims to assign schedules to blocks better, using the schedules generated by the graph-based method described in Chapter 2. The extension only add little extra time to the overall method, making sure the overall method is still useful for a decision support system. However, it only slightly improves the solutions.

We have in this chapter focused on improving the block assignment for the CMPBS, and as the results show there is still potential to improve, thus an idea for further research is to improve the way the block assignment is computed.

Chapter 6

Additional work for the Flexible Ship Loading Problem

6.1 Introduction

This chapter can be seen as an extension of the paper "Flexible ship loading problem with transfer vehicle assignment and scheduling" presented in Chapter 4 where the same problem is considered. In the following, a new mathematical model for the Flexible Ship Loading Problem (FSLP) is formulated, and a new hybrid heuristic is described to solve the problem. Both the model and heuristic improve the state-of-the-art. Lastly, a column generation algorithm is proposed. The column generation method can be seen as a starting point for a Branch-and-Price algorithm.

We believe the chapter, in its current form, is too unfocused and spans too wide to be published 'as is'. With additional work, we believe this chapter can be published as two standalone papers; One based on the revised model and the hybrid heuristic, and one focusing on a Branch-and-Price algorithm based on the column generation method here presented.

The rest of the chapter is organised as follows; Section 6.2 reiterate the problem definition in broader terms and presents a revised mathematical formulation and enhancements. Section 6.3 describes the hybrid heuristic, and Section 6.4 is a description of the column generation method. The results of these three methods are shown in Section 6.5 and lastly, Section 6.6 concludes on the work presented in this chapter.

6.2 Revised Flexible Ship Loading Problem model

In the FSLP a liner vessel docked at a port is considered. The containers destined for the port have been unloaded, and a set of containers are to be loaded on the ship. The liner provides the

terminal with a class-based stowage plan. A class based stowage specifies that a container of a specific *class* is to be loaded at a given position of the vessel. Here, a container class corresponds to the dimensions of the container, properties (reefer or dry cargo container), destination and weight of the container (e.g. light, medium or heavy). The exact weight of the container might not have an impact on the feasibility of the stowage plan, and thus it is sufficient to consider weight classes. The terminal is responsible for loading the vessel, following the stowage plan. The class-based stowage plan leaves much freedom for the terminal which they wish to exploit, to optimise their operations. The terminal might have multiple containers of the same type, and thus they want to determine which container goes where on the vessel, to optimise their workload while ensuring the vessel leaves as planned.

Consider a set of containers ($\mathcal{C}$) to be loaded in positions on the ship ($\mathcal{P}$) by a set of Quay Cranes (QCs) (the set $\mathcal{Q}$). The FSLP covers the assignment and scheduling of Transfer Vehicles (TVs) (set $\mathcal{S}$) to retrieve the containers from the yard and deliver in front of the QC. We assume that the loading order for each QC is determined beforehand, and is known. We use $p' \prec p$ to specify that the position p' will be loaded before p by the the same crane, and $p' \prec\prec p$ specifies that p' will be loaded immediately before p. The crane loading time is β, and thus there must be at least β time units between the loading of two successive positions.

The contract between the terminal and the liner specifies that with the amount of containers to be unloaded, the terminal is expected to finish loading the vessel at a given time - the expected finishing time (EFT). If this is not ensured the terminal must pay a penalty of γ for every time unit the vessel is delayed. The terminal must pay the operators operating the TVs, for the time they work. The time unit cost of this is α, and we assume this cost also includes equipment wear and tear and maintenance. The terminal aims to minimise the sum of these two costs.

The variables in this model are the same as in Chapter 4, i.e. $x^s_{cp} \in \{0, 1\}$ denote the assignment

$$x^s_{cp} = \begin{cases} 1, & \text{If container } c \in \mathcal{C} \text{ is loaded to position } p \in \mathcal{P} \text{ and picked up by TV } s \in \mathcal{S} \\ 0, & \text{Otherwise.} \end{cases}$$

and the variable $t^s_p \in \mathbb{N}$ denote the time at which TV $s \in \mathcal{S}$ drops the container for position $p \in \mathcal{P}$ in front of the QC. Also, let $Start_s$ and End_s be the start and end time of the service window for TV s. To account for the lateness of operation, define z as the makespan for loading the entire ship, and ΔEFT as the lateness of operations.

The model is similar to the model presented in Chapter 4, the main difference lies in how the drop off times (t^s_p) are handled. The model in Chapter 4 ensures that the drop off time is zero if a TV does not serve a position. This makes it necessary to have more *big-M* constraints deteriorating the LP bound. In the model presented here, if a TV, s, does not serve a position p, t^s_p is constrained to be greater than or equal to the time at which the TV dropped off the previous container. Compared to the model in Chapter 4, this model has fewer constraints and does not have as many *big-M* constraints, this should make the model easier to solve. We refer to this model as the revised flexible ship loading problem (R-FSLP) model.

Below, all the sets, variables, and parameters are summarised, and additional sets and parameters are introduced.

Sets:

$\mathcal{C}$	Set of containers
$\mathcal{C}_p$	Set of containers compatible with position $p \in \mathcal{P}$
$\mathcal{P}$	Set of positions to be loaded
$\mathcal{P}_c$	Set of positions combatible with container $c \in \mathcal{C}$
$\mathcal{P}_q$	Set of positions loaded by quay crane $q \in \mathcal{Q}$
$\mathcal{S}$	Set of transfer vehicles
$\mathcal{S}_p$	Set of transfer vehicle, that are available to serve position $p \in \mathcal{P}$
$\mathcal{S}_q$	Set of transfer vehicle, that are available to serve position quay crane $q \in \mathcal{Q}$
$\mathcal{Q}$	Set of quay cranes
q_p	The quay crane scheduled to load position $p \in \mathcal{P}$
K_p	The container-TV combinations that are compatible with position p $K_p = \{\langle c, s\rangle \in \mathcal{C} \times \mathcal{S} \mid c \in \mathcal{C}_p \wedge s \in \mathcal{S}_p\}$

Variables:

$x^s_{cp} \in \{0, 1\}$	1 if container $c \in \mathcal{C}$ is loaded to position $p \in \mathcal{P}$ and picked up by TV $s \in \mathcal{S}$, 0 otherwise.
$t^s_p \in \mathbb{Z}^+$	Time at which TV $s \in \mathcal{S}$ drop the container for position $p \in \mathcal{P}$ in front of the QC.
$Start_s \in \mathbb{Z}^+$	The time at which TV begins its service window
$End_s \in \mathbb{Z}^+$	The time at which TV stops working
$z \in \mathbb{Z}^+$	The time at which the loading of the entire ship is finished
$\Delta EFT \in \mathbb{Z}^+$	The tardiness of the operations.

Parameters:

τ_{cp}	The time needed for a TV to transport container c from its position in the yard to the vessel-position p. The time is assumed to be equal in both directions.
β	The crane loading time.
α	The cost of using a TV for one time-unit.
γ	The cost of exceeding the EFT by one time-unit.
EFT	The expected finishing time

With this, the problem can be modelled as seen below.

$$\texttt{Min Z} = \alpha \sum_{s \in \mathcal{S}} (End_s - Start_s) + \gamma \Delta EFT \tag{6.1}$$

`Subject to:`

$$\sum_{c \in \mathcal{C}_p} \sum_{s \in \mathcal{S}_p} x^s_{cp} = 1 \qquad \forall p \in \mathcal{P} \tag{6.2}$$

$$\sum_{p\in\mathcal{P}_c}\sum_{s\in\mathcal{S}_p} x^s_{cp} = 1 \qquad \forall c \in \mathcal{C} \tag{6.3}$$

$$x^s_{cp} = 0 \qquad \forall p \in \mathcal{P}, \langle c, s\rangle \notin K_p \tag{6.4}$$

$$t^s_{p'} + \sum_{c\in\mathcal{C}_p} 2\tau_{cp} x^s_{cp} \leq t^s_p \qquad \forall p \in \mathcal{P}, s \in \mathcal{S}_p, p' \prec\prec p \tag{6.5}$$

$$t^{s'}_{p'} + \beta \leq t^s_p + H\left(1 - \sum_{c\in\mathcal{C}_p} x^s_{cp}\right) \qquad \forall p \in \mathcal{P}, s, s' \in \mathcal{S}_{p'}, p' \prec\prec p \tag{6.6}$$

$$Start_s \leq t^s_p - \sum_{c\in\mathcal{C}_p} 2\tau_{cp} x^s_{cp} \qquad \forall p \in \mathcal{P}, s \in \mathcal{S}_p \tag{6.7}$$

$$t^s_p \leq End_s \qquad \forall q \in \mathcal{Q}, s \in \mathcal{S}_q, p = |\mathcal{P}_q| \tag{6.8}$$

$$z \geq End_s + \beta \qquad \forall s \in \mathcal{S} \tag{6.9}$$

$$\Delta EFT \geq z - EFT \tag{6.10}$$

$$x^s_{cp} \in \{0, 1\} \qquad \forall c \in \mathcal{C}, p \in \mathcal{P}, s \in \mathcal{S} \tag{6.11}$$

$$t^s_p \in \mathbb{Z}^+ \qquad \forall p \in \mathcal{P}, s \in \mathcal{S} \tag{6.12}$$

$$Start_s, End_s \in \mathbb{Z}^+ \qquad \forall s \in \mathcal{S} \tag{6.13}$$

$$z, \Delta EFT \in \mathbb{Z}^+ \tag{6.14}$$

Objective (6.1) minimises the cost for loading the ship. Constraints (6.2) and (6.3) ensures that a container is loaded in every position, and all containers are loaded on the ship. Constraint (6.4) ensures that no incompatible assignment is made. Constraints (6.5) and (6.6) updates the time if a TV serves a position. Here the position p' is the position served immediately before position p according to the loading order for the crane q_p. Specifically (6.5) ensure that there is enough time for the TV to pickup the assigned container, and deliver it in front of the QC, where as (6.6) make sure the crane loading time is satisfied. Constraint (6.6) is a linearised version of the following non-linear constraint.

$$\sum_{c\in\mathcal{C}_p} x^s_{cp} = 1 \Rightarrow \max_{s'\in\mathcal{S}_{p'}}(t^{s'}_{p'}) + \beta \leq t^s_p \qquad \forall p \in \mathcal{P}, s \in \mathcal{S}_p, p' \prec\prec p$$

The constraint have been linearised using the *big-M*-method, where H is the maximum time the loading can take. Constraints (6.7) and (6.8) sets the start and end time for the TVs. The makespan and the tardiness is calculate in constraints (6.9) and (6.10). Lastly (6.11) - (6.14) defines the variable domains.

In Chapter 4, a number of valid inequalities were added to the base model, which helped to improve the lower bound. The valid inequality that helped the most was the following

$$End_s - Start_s \geq \sum_{c\in\mathcal{C}}\sum_{p\in\mathcal{P}_c} 2\tau_{cp} x^s_{cp} \qquad \forall s \in \mathcal{S} \tag{6.15}$$

This ensures that the total time a TV works is greater than or equal to the total transportation time needed for that TV. The combination of (6.5), (6.7) and (6.8) of the R-FSLP model already ensures that the valid inequality (6.15) is satisfied, and will thus only add extra complexity to the model if it is added. However, the valid inequalities (4.17) and (4.19) can be used here as well. Instead of the symmetry breaking constraint (4.20) based on the working time of the vehicles we here define a symmetry breaking constraint based on the container assignment. The container

assignment variables are the main variables in the model, and we therefore expect a symmetry breaking constraint on those variables to be better than a symmetry breaking constraint on auxiliary variables. The symmetry breaking constraint is the following

$$\sum_{p\in\mathcal{P}_q}\sum_{c\in\mathcal{C}_p} x_{cp}^{s} \leq \sum_{p\in\mathcal{P}_q}\sum_{c\in\mathcal{C}_p} x_{cp}^{s+1} \qquad \forall q\in\mathcal{Q}, s\in\mathcal{S}_q\setminus\{s_q^{|\mathcal{S}_q|}\} \tag{6.16}$$

Here it is enforces, that of two vehicles for the same QC, the one with the higher index cannot be assigned more containers than the one with the lower index.

We use R-FSLP+ to denote the model R-FSLP with the extra constraints (4.17), (4.19) and (6.16).

6.3 A Hybrid heuristic for the FSLP

In Chapter 4 a GRASP heuristic is presented for the FSLP. The heuristic splits the decisions in two. First, the *service times* for the TVs are chosen after which container/position are assigned to TVs following the assigned service times. The adaptive control method governs how the service times are determined. In each iteration, a solution is built from scratch, and the only information shared between two iterations is how to select the service times. To extend the algorithm presented in Chapter 4 we will present a hybrid heuristic, combining a genetic algorithm with the GRASP method. The genetic algorithm will combine two solutions, and this way information from a previous solution is reused in a later iteration. The GRASP method is used to provide diversity to the population considered in the genetic algorithm.

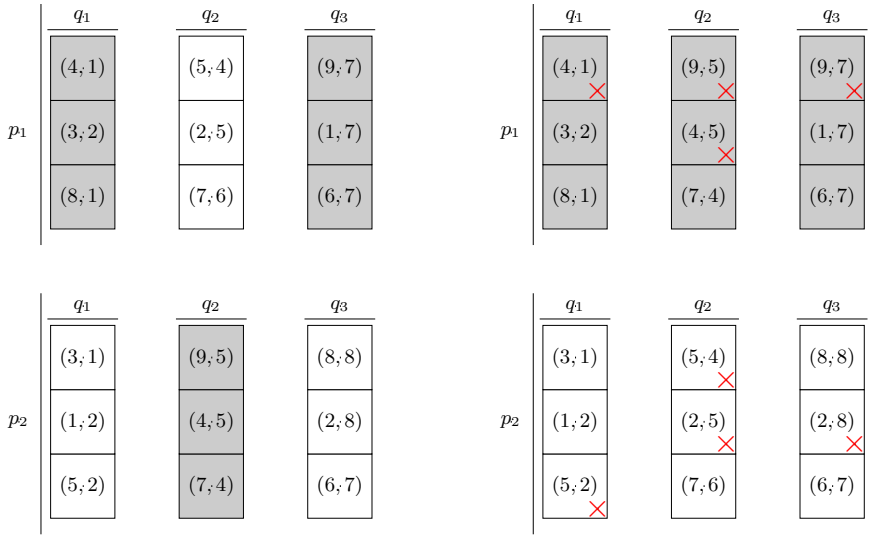
A feasible solution to the FSLP can be seen as a combination of *partial solutions*, one for each QC. In this context, a partial solution (also denoted QC solution) only assigns containers/TVs to the positions to be served by a single QC. In a full solution, some of the partial solutions might be bad, while others might be good. Thus combining two solutions using a genetic algorithm could result in a better overall solution

Using a *crossover* method we will combine partial solutions from two *parent* solutions into an *offspring* solution. Consider a population of $|\Pi|$ *individuals*, (i.e solutions), the proposed genetic algorithm randomly selects 4 individuals, x_1, x_2, x_3, x_4. Based on these 4 solutions two parents solutions are selected

$$p_1 = \underset{x\in\{x_1\cup x_2\}}{\arg\min}\,(f(x))\,, \qquad p_2 = \underset{x\in\{x_3\cup x_4\}}{\arg\min}\,(f(x)) \tag{6.17}$$

where $f(x)$ is the fitness of the individual x. In eq. (6.17) the most fit of the first two individuals is selected as the first parent, and the most fit between the last two individuals is the second parent. Based on the two parents, two offsprings are made. For the crossover method, let $\Omega_1 \subset \mathcal{Q}$ be a subset of cranes. For all $q \in \Omega_1$ the QC solution for q will be copied from parent p_1 to the first offspring. For all $q \in \mathcal{Q}\setminus\Omega_1$ the QC solutions from p_2 is used. In a similar fashion let $\Omega_2 = \mathcal{Q}\setminus\Omega_1$ be the cranes for which the QC solution from p_1 should be copied to the second offspring. Figure 6.1 illustrates the crossover method. In the example there are 3 QCs and all of them has to load 3 positions. Each of the QCs have 3 TVs assigned to them, numbered $1,2,3$ for q_1, $4,5,6$ for q_2, and $7,8,9$ for q_3. The numbers are of the format (c,s), and represent a

container c to be picked up by TV s, to be loaded to the position. Figure 6.1a shows two parents solutions, and Figure 6.1b are the offspring when $\Omega_1 = \{1, 3\}$ and $\Omega_2 = \{2\}$.



(a) Parent solutions *(b) Offspring solutions*

Figure 6.1: Illustration of the crossover method with $\Omega_1 = \{1, 3\}$ and $\Omega_2 = \{2\}$

The crossover method does not automatically result in a feasible solution. In Figure 6.1b the crosses represents the containers that have been scheduled for multiple positions. To restore feasibility of the offspring solutions we use a *mutation method*. The mutation method identifies and removes all occurrences of containers that are scheduled for multiple positions. Along with the containers not scheduled, the removed containers are then distributed randomly to the, now unserviced compatible positions, and thus ensuring a feasible solution. While doing so, the assignment of vehicles to position remains unchanged, but the scheduled times might change due to new travelling times.

By using the described crossover method, the population will quickly converge, thus limiting the applicability of this approach. To counter this, we combine the genetic algorithm with the GRASP heuristic described in Chapter 4. The parameter ρ describes the probability of generating a new solution instead of using the crossover method. The GRASP method is then used for n_{GRASP} iterations, and the best solution is included in the population. The adaptiveness of the GRASP method is preserved, and the quality of the solutions generated should thus increase during the execution of the hybrid algorithm. Here n_{GRASP} should be kept low to ensure the execution time does not increase considerably. The GRASP method is also used to initialize the population Π.

The overall hybrid algorithm can be seen in Algorithm 7. The hybrid algorithm receives the three parameters, $|\Pi|$, ρ and n_{GRASP} as input and line 1 initialises the population Π by using the GRASP method. Here $|\Pi|$ iterations are performed, and each solution is included in the population. The while loop in lines 2-24 runs until it is terminated. The termination criterion used here is based on the number of iterations n. Lines 3-7 selects the two parents solutions and generates the sets Ω_1 and Ω_2 as previously described. Lines 8-23 generates two new solutions and include those in the population. Based on the value of r, line 10 determines if the crossover method or the GRASP method should be used. In line 11 the new solution is made based on the

Algorithm 7 `Hybrid Algorithm`

Input: $|\Pi|$, ρ, n_{GRASP}

$\Pi \leftarrow$ `InitializePopulation`$(|\Pi|)$
while !`Terminate` **do**
  x_1, x_2, x_3, $x_4 \leftarrow$ 4 random distinct individuals from Π
  $p_1 \leftarrow \underset{x \in \{x_1 \cup x_2\}}{\arg\min} \; (f(x))$
  $p_2 \underset{x \in \{x_3 \cup x_4\}}{\arg\min} \; (f(x))$
  $\Omega_1 \leftarrow$ Uniform random set of cranes. $1 \leq |\Omega_1| < |Q|$
  $\Omega_2 \leftarrow Q \setminus Q_1$
  for all $\omega \in \{\Omega_1 \cup \Omega_2\}$ **do**
    $r \leftarrow$ Uniform random number in $[0; 1]$
    if $r < 1 - \rho$ **then**
      $c \leftarrow$ `Crossover`(p_1, p_2, ω)
      `Mutate`(c)
      `VehicleReassignment`(c)
    else
      $c \leftarrow$ `GenerateSolution`(n_{GRASP})
    end if
    if !`FitnessValueInPopulation`$(f(c))$ **then**
      y_1, $y_2 \leftarrow$ 2 random distinct individuals from Π
      $d \leftarrow \underset{y \in \{y_1 \cup y_2\}}{\arg\max} \; (f(y))$
      Π.`remove`(d)
      Π.`add`(c)
    end if
  end for
end while
return $\underset{x \in \Pi}{\arg\min} \, (f(x))$

two parents solution, and the crossover set ω, the `Mutate` method in 12 ensures the feasibility as previously described. The `VehicleReassignment` in line 13 is an improvement method based on reassigning the TVs. This method is also used as part of the GRASP method, and is described in Chapter 4. If the crossover method is not used then a solution is generated by using the GRASP method in line 15. Lines 17-22 updates the population. Here `FitnessValueInPopulation`$(f(c))$ checks if the population contains an individual with the fitness $f(c)$, if so the solution c is not included in the population. Lines 18-20 selects two random solutions from the population and removes the worst of those from the population. Instead, the new individual c is added. Lastly, line 25 returns the best solution in the population as the best-found solution.

6.4 Column Generation for the FSLP

We now present a column generation approach for FSLP. The aim of this is to provide a starting point which further research can be built on, to provide better lower bounds for the FSLP, or ultimately solve the problem using Branch-and-Price.

6.4.1 The master problem

An alternative way to model the problem is to consider all the feasible *assignments* for a single vehicle. Here an assignment describes the containers and positions a vehicle will serve, and at which time. In this problem, all times (β, EFT and all τ_{cp} parameters) are assumed to be integer. Thus there is no accuracy loss by discretising the times in time steps of 1 unit. Let $\mathcal{T}$ be the set of all discretised times, from 0 to H with steps of 1 unit. Also, define $\Omega(q)$ as the set of all feasible TV assignments for a QC q. For a given assignment a, let a_{ac} and b_{ap} describe the containers and positions the assignment serves, i.e. a_{ac} is 1 if assignment a assigns container c to a position and 0 otherwise, and b_{ap} is 1 if assignment a assigns a container to position p and 0 otherwise. If an assignment assigns a container to a position p, then t_{pa} denotes the time at which it is scheduled to be delivered in front of the QC. Given an assignment, we can easily calculate the service length (ν_q) and the time at which it ends the operation. Specifically, let $End_a^t = 1$ if assignment a ends at time $t \in \mathcal{T}$. To model the problem, introduce y_{qa} as a binary variable.

$$y_{qa} = \begin{cases} 1, & \text{If quay crane } q \text{ uses assignment } a \\ 0, & \text{Otherwise.} \end{cases}$$

and let the variable $z^t \in \{0,1\}$ be 1 if the operation is finished at time t. With this and the notation introduced in Section 6.2, the problem can be modelled as follows.

$$\texttt{Min Z} = \alpha \sum_{q \in \mathcal{Q}} \sum_{a \in \Omega(q)} \nu_a y_{qa} + \gamma \Delta EFT \tag{6.18}$$

`Subject to:`

$$\sum_{a\in\Omega(q)} y_{qa} \leq |\mathcal{S}_q| \qquad \forall q \in \mathcal{Q} \tag{6.19}$$

$$\sum_{a\in\Omega(q)} a_{ac} y_{qa} = 1 \qquad \forall c \in \mathcal{C} \tag{6.20}$$

$$\sum_{a\in\Omega(q)} b_{ap} y_{qa} = 1 \qquad \forall q \in \mathcal{Q}, p \in \mathcal{P}_q \tag{6.21}$$

$$\sum_{a\in\Omega(q)} t_{p'a} b_{ap'} y_{qa} + \beta \leq \sum_{a\in\Omega(q)} t_{pa} b_{ap} y_{qa} \qquad \forall q \in \mathcal{Q}, p \in \mathcal{P}_q, p' \prec\prec p \tag{6.22}$$

$$\sum_{t'=t}^{|T|} |\mathcal{S}_q| z^{t'} \geq \sum_{a\in\Omega(q)} \sum_{t'=t}^{|T|} End_a^{t'} y_{qa} \qquad \forall q \in \mathcal{Q}, t \in \mathcal{T} \tag{6.23}$$

$$\Delta EFT \geq \sum_{t\in\mathcal{T}} t z^t - EFT \tag{6.24}$$

$$y_{qa} \in \{0,1\} \qquad \forall q \in \mathcal{Q}, a \in \Omega(q) \tag{6.25}$$

$$z^t \in \{0,1\} \qquad \forall t \in \mathcal{T} \tag{6.26}$$

$$\Delta EFT \in \mathbb{Z}^+ \tag{6.27}$$

The objective (6.18) is equivalent to (6.1), and thus minimises the cost for the terminal. The first constraint (6.19) guarantees that we only selects as many assignments for a crane as we have vehicles available. Constraints (6.20) and (6.21) corresponds to (6.2) and (6.3) of Section 6.2, and ensures that all containers and positions are served exactly once. Constraint (6.22) calculates the time at which two consecutive positions are served, and enforces that the time between is greater than or equal to β. Constraint (6.23) ensures that if the operations for a chosen assignment ends at time t then at least one $z^{t'} = 1$ for $t' \in \{t \dots |\mathcal{T}|\}$ and constraint (6.24) combined with the objective ensures that $z^t = 1$ for the earliest time at which the total operation is finished. Eq. (6.24) also sets the variable ΔEFT. Lastly constraints (6.25)-(6.27) defines the variable domains.

The drawback of the model (6.18)-(6.27) is that it requires the enumeration of all feasible assignments. This makes the model computationally intractable, as the number of feasible assignments grows exponentially. However, only a small number of the all the variables will be used in an optimal solution. Therefore, instead of considering the full subset of the assignments, we can consider a smaller subset of generated assignments $\Omega_{gen}(q)$, and generate more assignments along the way.

This is exactly the idea of column generation, and for this we make a continuous relaxation of the y_{qa} and z^t variables. Additionally, we define δ_q as the dual variables associated with constraint (6.19), and let θ_c be the dual variables for constraint (6.20), λ_p for (6.21), π_p for (6.22) and lastly the dual variables for (6.23) are denoted μ_{qt}. These dual variables will be used to calculate the reduced cost of a variable in the pricing problem.

6.4.2 The pricing problem

The pricing problem generates assignments, and thus adds variables to the master problem. The purpose of the pricing problem is two-fold: First, if any negative reduced costs assignments exist it should identify at least one to add to the master. Secondly, prove that no reduced costs assignments exist. In our case, we will solve $|Q|$ pricing problems in every iteration, one for every QC. The pricing problem can be modelled as a mixed integer programming problem. For this let the variable $x_{cp} \in \{0,1\}$ denote if container c is assigned to position p, and let $w_p^t \in \{0,1\}$ be 1 if position p is served at time t, 0 otherwise. Also, let $End^t \in \{0,1\}$ be 1 if the assignment end at time t, 0 otherwise. Lastly define $Start$ as the start time, and z as the end time of the operation. With this and the notation introduced in Section 6.2, the pricing problem for the QC q can be modelled as seen below.

$$\texttt{Min Z} = \alpha(z - Start) - \delta_q - \sum_{p \in \mathcal{P}_q} \sum_{c \in \mathcal{C}_p} (\theta_c + \lambda_p) x_{cp} - \sum_{t \in \mathcal{T}} \sum_{t'=0}^{t} \mu_{qt'} End^t - \sum_{p \in \mathcal{P}_q} \sum_{t \in \mathcal{T}} (tw_{p'}^t - tw_p^t)\pi_p \tag{6.28}$$

Subject to:

$$\sum_{p \in \mathcal{P}_q} x_{cp} \leq 1 \qquad \forall c \in \mathcal{C} \tag{6.29}$$

$$\sum_{c \in \mathcal{C}_p} x_{cp} \leq 1 \qquad \forall p \in \mathcal{P}_q \tag{6.30}$$

$$x_{cp} = 0 \qquad \forall p \in \mathcal{P}_q, c \notin \mathcal{C}_p \tag{6.31}$$

$$\sum_{c \in \mathcal{C}_p} x_{cp} = \sum_{t \in \mathcal{T}} w_p^t \qquad \forall p \in \mathcal{P}_q \tag{6.32}$$

$$\sum_{t \in \mathcal{T}} tw_{p'}^t + \sum_{c \in \mathcal{C}_p} \max\left((p - p')\beta, 2\tau_{cp}\right) x_{cp} \leq \sum_{t \in \mathcal{T}} tw_p^t + H(1 - \sum_{c \in \mathcal{C}_p} x_{cp}) \qquad \forall p \in \mathcal{P}_q, p' \prec p \tag{6.33}$$

$$\sum_{t \in \mathcal{T}} tw_p^t - \sum_{c \in \mathcal{C}_p} 2\tau_{cp} x_{cp} + H(1 - \sum_{c \in \mathcal{C}_p} x_{cp}) \geq Start \qquad \forall p \in \mathcal{P}_q \tag{6.34}$$

$$z \geq \sum_{t \in \mathcal{T}} tw_p^t \qquad \forall p \in \mathcal{P}_q \tag{6.35}$$

$$\sum_{t \in \mathcal{T}} tEnd^t \geq z \tag{6.36}$$

$$\sum_{t \in \mathcal{T}} End^t \leq 1 \tag{6.37}$$

$$Start \leq z \tag{6.38}$$

$$x_{cp} \in \{0,1\} \qquad \forall c \in \mathcal{C}, p \in \mathcal{P}_q \tag{6.39}$$

$$w_p^t \in \{0,1\} \qquad \forall p \in \mathcal{P}_q, t \in \mathcal{T} \tag{6.40}$$

$$End^t \in \{0,1\} \qquad \forall t \in \mathcal{T} \tag{6.41}$$

$$Start, z \in \mathbb{Z}^+ \tag{6.42}$$

The objective (6.28) minimises the reduced costs, and thus guarantees to find a negative reduced cost assignment if one exists. Otherwise, it will prove that no reduced costs assignments exist. The constraints (6.29) and (6.30) guarantees that a container and position is served at most once. In (6.31) it is ensured that no incompatible assignment is made and (6.32) enforces that if a container is assigned a position, a time must be chosen as well. Constraint (6.33) ensures that there is enough time between two served positions. Here p' is a position before p in the loading order, and $(p' - p)$ is the number of positions between the two positions. Constraint (6.34) set the $Start$ variable, and constraint (6.35) sets the z variable. Constraint (6.36) links the End^t variables with the z variable, and constraint (6.37) ensures at most one of the End^t variable is one. Constraint (6.38) is needed in the case where no containers are assigned to a position, to ensure that the term $\alpha(z - Start)$ in the objective function does not become negative. Constraints (6.39)-(6.42) defines the domain for the variables.

Solving the model (6.28)-(6.42) $|Q|$ times per iteration will be computationally expensive, consequently slowing down the overall method. However, with minor modifications to the problem, it can be solved as a simple shortest path problem in a time-expanded graph.

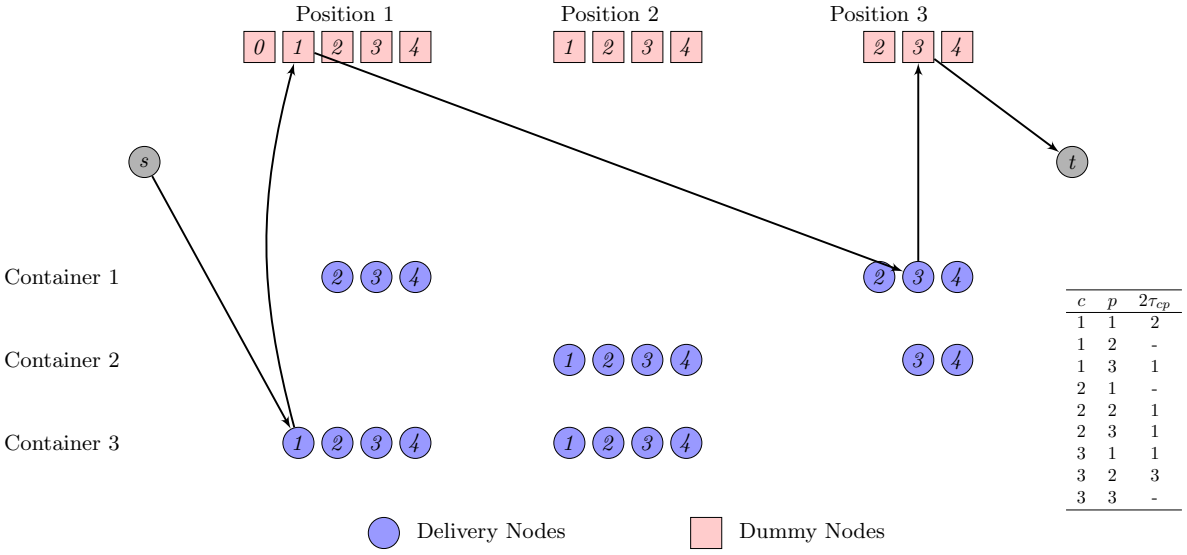
6.4.3 The pricing problem as a shortest path problem

Constraint (6.29) of the pricing problem makes certain that an assignment only schedules a container to at most a single position. However, if we allow assignments to assign the same container to multiple positions, the pricing problem can be solved as a simple shortest path problem.

Let G be a weighted directed acyclic graph with $V(G)$ as the vertex set and $E(G)$ as the set of edges. A vertex symbolizes a container c to be loaded onto position p at time t. Thus a vertex i can be described by the tuple $\langle p, c, t \rangle$. Figure 6.2 shows an example graph with 1 QC, 3 positions and containers, $H = 4$ and $\beta = 1$. Figure 6.2 illustrate the vertices, and the labels on top and to the left describe which position/container the node is for, and the number within the node is the time associated with the node. As can be seen from the graph, not all containers are compatible with all positions. The time it takes to go from the QC to the containers yard position and back to the QC is the time $2\tau_{cp}$ shown in the table in Figure 6.2.

Besides the source and the terminal, we define two sets of vertices; *delivery vertices* and *dummy vertices*. The delivery vertices are for scheduled deliveries, and the dummy vertices help to satisfy the constraints of the problem. For a position p there is a set of vertices for every container c that is compatible with position p. For a given position-container combination we calculate the earliest possible time this container c can be delivered for position p. For this, let q_f be the first position served by the considered QC q. Thus $q_f - p$ is the number of positions the crane will load before loading position p, and the earliest possible time the container can be loaded is thus $\max((p - q_f)\beta, 2\tau_{cp})$. Here we both consider the crane loading time and the travelling time to retrieve the considered container. Therefore the set of delivery nodes can be defined as follows.

$$V_{Delivery} = \{i = \langle p, c, t \rangle \mid p \in \mathcal{P}_q \ \wedge \ c \in \mathcal{C}_p \ \wedge \ t \in \mathbb{Z}^+ \ \wedge \ \max((p - q_f)\beta, 2\tau_{cp}) \leq t \leq H\} \tag{6.43}$$



c	p	$2\tau_{cp}$
1	1	2
1	2	-
1	3	1
2	1	-
2	2	1
2	3	1
3	1	1
3	2	3
3	3	-

Figure 6.2: Vertex illustration. The numbers within the nodes describes the time the node is associated with, and the table describe the travelling times for the position/container combinations.

For every delivery node, there is one single out-going edge to the dummy node for the same time and position. There is no container attached to the dummy nodes, and they can thus be described by the tuple $\langle p, \emptyset, t \rangle$. The out-going edges of the dummy nodes ensure that there is sufficient time between two deliveries of an assignment. The dummy nodes can be defined as follows.

$$V_{Dummy} = \{i = \langle p, \emptyset, t \rangle \mid p \in \mathcal{P}_q \ \wedge \ t \in \mathbb{Z}^+ \ \wedge \ (p - q_f)\beta \leq t \leq H\} \tag{6.44}$$

Finally, we have a source, s, and a terminal t. Hence the full set of vertices is

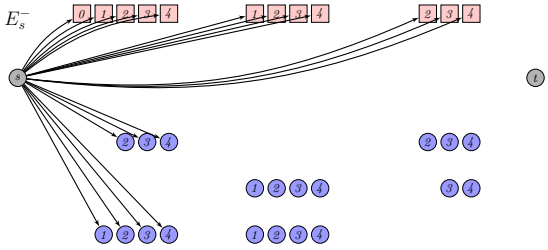
$$V(G) = s \cup t \cup V_{Delivery} \cup V_{Dummy}$$

The full set of edges $E(G)$ can be split up into 4 sets of edges. First, the edges leaving the source node (E_s^-). Second, the edges leaving the dummy nodes excluding the ones to the terminal vertex (E_{Dummy}^-). Third, the edges leaving the delivery nodes ($E_{Delivery}^-$), and lastly the set of edges entering the terminal vertex (E_t^+). These edges are illustrated in Figure 6.3. In Figure 6.3b only edges from the shown dummy nodes are illustrated. The edges from the set E_{Dummy}^- ensures there that there is sufficient time between two containers scheduled in the assignment.

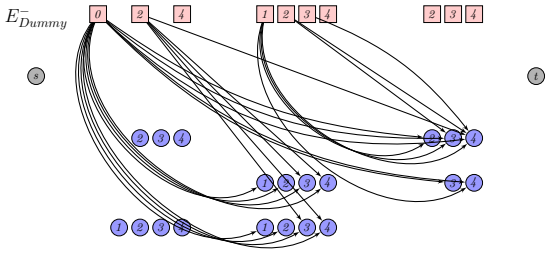
From the source vertex there are edges going out to all the dummy nodes, and all the delivery nodes for the first served position.

$$E_s^- = \{(s, i) | i \in V_{Delivery}(q_f) \vee i \in V_{Dummy}\} \tag{6.45}$$

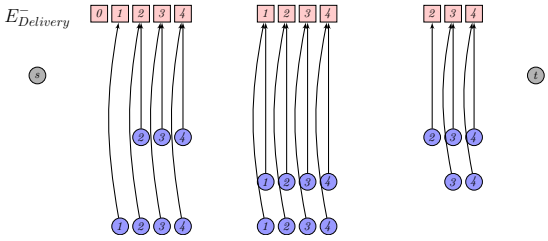
Here $V_{Delivery}(q_f)$ are the set of delivery vertices for the position q_f. A dummy node has edges going to all *compatible* delivery nodes. In this context, two nodes are compatible if the time difference is sufficiently large, considered the travelling time and crane loading time. Let p_i be the position for node i and t_i the time for node i, with this the set E_{Dummy}^- can be defined as



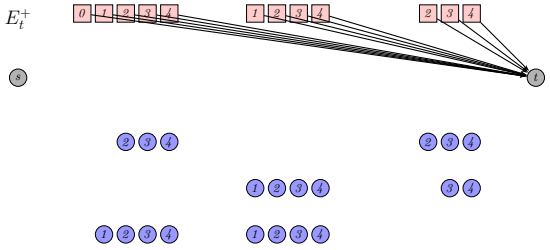
(a) Illustration of the edges from the set E_s^-



(b) Illustration of the edges from the set E_{Dummy}^-. Only the edges from the shown dummy nodes are illustrated.



(c) Illustration of the edges from the set $E_{Delivery}^-$



(d) Illustration of the edges from the set E_t^+

Figure 6.3: Illustration of the edges

follows.

$$E_{Dummy}^- = \{(i,j), i \in V_{Dummy}, j = \langle p, c, t\rangle | p > p_i \wedge c \in \mathcal{C}_p \wedge t_i + \max(2\tau_{cp}, (p-p_i)\beta) \leq t \leq H\} \tag{6.46}$$

As mentioned earlier, all delivery nodes only have a single out-going edge, namely to the associated dummy node.

$$E_{Delivery}^- = \{(i,j), i \in V_{Delivery}, j = \langle p, \emptyset, t\rangle | p = p_i, t = t_i\} \tag{6.47}$$

From all the dummy nodes there are an edge to the terminal vertex, and E_t^+ can thus be defined as follows.

$$E_t^+ = \{(i,t) | i \in V_{Dummy}\} \tag{6.48}$$

The full set of edges is then

$$E(G) = E_s^- \cup E_{Dummy}^- \cup E_{Delivery}^- \cup E_t^+$$

The $s-t$-path shown in Figure 6.2 corresponds to an assignment where position 1 is served at time 1 with container 3 and position 3 is served at time 3 with container 1.

The weight of the edges depends on the dual variables from the master, and the cost of any $s-t$-path must coincide with the reduced cost of the corresponding variable. For this, we split the weight of an edge up into four sub parts. First, a cost for moving forward in the time dimension

$(w(\alpha))$. Second, one for picking up containers and servicing a position $(w(\theta, \lambda))$. Third, one depending on the time a position is served $(w(\pi))$, and lastly one depending on the end time $(w(\mu))$. The total weight of an edge can be written as

$$w = w(\alpha) + w(\theta, \lambda) + w(\pi) + w(\mu)$$

Table 6.1 breaks down the value of each of the four sub parts, depending on the type of edge. Here the set E_s^- if broken in two, one for the edges going to delivery nodes, and one for the edges going to dummy nodes. Let R be the set of edges in the shortest $s-t$-path. Then $\sum_{e \in R} e(w(\alpha))$ will be the time cost for the assignment, and thus corresponds to the first term in (6.28). Here $e(w(\alpha))$ is the value of $w(\alpha)$ for the edge e. $\sum_{e \in R} e(w(\theta, \lambda))$ will coincide with the third term, and $\sum_{e \in R} e(w(\mu))$ will be the fourth term. Lastly $\sum_{e \in R} e(w(\pi))$ will be equal to the fifth term. The reduced cost of the corresponding variable will be

$$r = \sum_{e \in R} e(w) - \delta_q$$

if $r < 0$ the variable corresponding to the path has negative reduced cost and is then added to the master problem.

Table 6.1: Breakdown of the edge weights.

	$E^-_{s,delivery}$	$E^-_{s,dummy}$	E^-_{Dummy}	$E^-_{Delivery}$	E^+_t
$w(\alpha)$	αt_j	0	$\alpha(t_j - t_i)$	0	0
$w(\theta, \lambda)$	$-(\theta_{c_j} + \lambda_{p_j})$	0	$-(\theta_{c_j} + \lambda_{p_j})$	0	0
$w(\pi)$	0	0	0	$-(t_i \pi_{p_i} - t_i \pi_{p_i+1})$	0
$w(\mu)$	0	0	0	0	$-\sum_{t=0}^{t_i+\beta} \mu_{qt}$

The graph as described here is a directed acyclic graph, and thus the shortest path can be calculated in linear time by sorting the vertices in a topologically order, before using Dijkstra's algorithm.

6.5 Results

In this section, we report the results of the presented models and methods. The methods presented have been tested using a 2.30 GHz Intel Xeon E5 Processor. All models are solved using CPLEX version 12.7.

6.5.1 Results for the R-FSLP model

To compare the results from the R-FSLP model with the FSLP+ model from Chapter 4, it has been tested under similar conditions; A time limit of 3 hours is imposed, and the models are run with four threads.

Table 6.2 and Table 6.3 shows the result from the R-FSLP model, and its extensions. The first columns describe the instance characteristics, with $|C|$ being the number of containers, $|CT|$

the number of container types and $|Q|$ the number of QCs. D is the density, which describes the distance between the containers in the yard. The densities are Less Dense (LD), Scattered (S) and Uniform (U). For all of the tested models, the table reports the solution in the root node (x^{Root}), the final lower bound (x^{LB}), the value of the final solution (x^{UB}). In the tables, $Gap(x^{UB})$ is the relative difference between the lower bound and final solution and $t(s)$ is the time spent in seconds. The values for the best found bound, and the best-found solution is written in **bold**.

Table 6.2 reports the results for the R-FSLP and R-FSLP+ model and compares with the results from the FSLP+ model from Chapter 4. Comparing R-FSLP and FSLP+, it is clear that R-FSLP is a better model than FSLP+. The gap is substantially improved in almost all of the test instances. The improvement is both due to improved lower bounds and better solutions. Comparing R-FSLP with R-FSLP+, it can be seen that adding the extra constraints ((4.17), (4.19), and (6.16)) does not improve the performance, but the contrary. A reason for this could be that the added complexity outweighs the benefits with regards to bound improvements and generated cuts.

Table 6.3 evaluate the benefit of each of the valid inequalities. Here we remove constraints from the extended model (R-FSLP+) one at a time. Comparing with the results from the extended model we can then evaluate the benefit of adding the constraints. Table 6.3 shows that the symmetry breaking constraint (6.16) deteriorates the model. Removing the constraint, the performance of the model is comparable to the simple model R-FSLP.

Table 6.2: Performance of enhancements. '–' symbolises that no feasible solution were found within the time limit, and '†' is used for the instances where the execution was terminated due to insufficient memory.

				FSLP+					R-FSLP					R-FSLP+				
$\|C\|$	$\|CT\|$	$\|Q\|$	D	x^{Root}	x^{LB}	x^{UB}	$Gap(x^{UB})$	$t(s)$	x^{Root}	x^{LB}	x^{UB}	$Gap(x^{UB})$	$t(s)$	x^{Root}	x^{LB}	x^{UB}	$Gap(x^{UB})$	$t(s)$
60	10	2	*LD*	1500	1500	1775	15.5%	10800	1530	**1563**	**1665**	6.10%	10800	1530	1560	1710	8.77%	10800
60	10	2	*S*	1020	**1020**	1030	1.0%	10800	1020	**1020**	**1020**	0.00%	50	1020	**1020**	**1020**	0.00%	208
60	10	2	*U*	1640	1640	1920	14.6%	9004†	1652	**1680**	**1770**	5.08%	10800	1652	**1680**	1825	7.95%	10800
60	25	2	*LD*	2030	2030	2345	13.4%	10800	2060	2199	2265	2.89%	10800	2060	2220	2250	1.33%	10800
60	25	2	*S*	1360	1360	1470	7.5%	6215†	1360	1360	1410	3.55%	10800	1360	1360	**1395**	2.51%	10800
60	25	2	*U*	1490	**1490**	1520	2.0%	10800	1490	**1490**	**1490**	0.00%	379	1490	**1490**	**1490**	0.00%	227
			Average				9.0%	9737				2.94%	7271				3.43%	7273
240	20	2	*LD*	7850	7850	14795	46.9%	10800	7880	**7880**	11690	32.59%	10800	7880	**7880**	11065	28.78%	10800
240	20	2	*S*	4440	**4440**	8225	46.0%	10800	4440	**4440**	**4950**	10.30%	10800	4440	**4440**	5040	11.90%	10800
240	20	2	*U*	6720	**6720**	11765	42.9%	10800	6720	**6720**	**8425**	20.24%	10800	6720	**6720**	9380	28.36%	10800
240	60	2	*LD*	8230	8230	11035	25.4%	10800	8260	**8260**	10240	19.34%	10800	8260	**8260**	10860	23.94%	10800
240	60	2	*S*	5280	**5280**	6555	19.5%	10800	5280	**5280**	6555	19.45%	10800	5280	**5280**	**6535**	19.20%	10100†
240	60	2	*U*	7250	7250	10845	33.1%	10800	7580	**7610**	10235	25.65%	10800	7580	**7610**	**9435**	19.34%	10600†
			Average				35.6%	10800				21.26%	10800				21.92%	10650
500	20	4	*LD*	14390	14390	-	-	10800	14420	**14420**	21445	32.76%	10800	14420	**14420**	20820	30.74%	10800
500	20	4	*S*	8250	**8250**	-	-	10800	8250	**8250**	11315	27.09%	10800	8250	**8250**	11775	29.94%	10800
500	20	4	*U*	14350	14350	-	-	10800	14380	**14410**	**19120**	24.63%	10800	14380	14380	22180	35.17%	10800
500	60	4	*LD*	14460	14460	-	-	10800	14490	14490	19660	26.30%	10800	14490	**14520**	24740	41.31%	10800
500	60	4	*S*	11840	11840	-	-	10800	11840	**11840**	15110	21.64%	10800	11840	**11840**	14035	15.64%	10800
500	60	4	*U*	15500	15500	264170	94.1%	10800	15530	15540	20020	22.38%	10800	15530	15530	21680	28.37%	10800
500	100	4	*LD*	15390	15390	-	-	10800	15420	**15450**	19205	19.55%	10800	15420	**15450**	19355	20.18%	10800
500	100	4	*S*	9490	**9490**	-	-	10800	9490	**9490**	10610	10.56%	10800	9490	**9490**	**10575**	10.26%	10800
500	100	4	*U*	16230	16230	22625	28.3%	10800	16260	**16290**	19390	15.99%	10800	16260	**16290**	20030	18.67%	10800
			Average				61.2%	10800				22.32%	10800				25.59%	10800
1000	20	4	*LD*	†	†	†	†	†	21970	**21990**	**48185**	54.36%	10800	21970	**21990**	-	-	10800
1000	20	4	*S*	†	†	†	†	†	14570	**14570**	-	-	10800	14570	**14570**	-	-	10800
1000	20	4	*U*	†	†	†	†	†	25950	25950	63125	58.89%	10800	25940	**25980**	**58680**	55.73%	10800
1000	60	4	*LD*	27070	27070	-	-	10800	27080	**27100**	63770	57.50%	10800	27100	**27100**	-	-	10800
1000	60	4	*S*	17950	17950	-	-	10800	17950	**17950**	**50210**	64.25%	10800	17950	**17950**	56485	68.22%	10800
1000	60	4	*U*	40230	40230	-	-	10800	40260	**40260**	405115	90.06%	10800	40260	**40260**	402555	90.00%	10800
1000	100	4	*LD*	28740	28740	-	-	10800	28770	**28770**	-	-	10800	28770	**28770**	308400	90.67%	10800
1000	100	4	*S*	17840	17840	-	-	10800	17840	**17840**	264100	93.24%	10800	17840	**17840**	-	-	10800
1000	100	4	*U*	32180	32180	-	-	10800	32210	**32210**	86965	62.96%	10800	32210	**32210**	**66580**	51.62%	10800
			Average				-	10800				68.75%	10800				71.25%	10800

Table 6.3: Performance of enhancements. '–' symbolises that no feasible solution were found within the time limit, and '†' is used for the instances where the execution was terminated due to insufficient memory.

				R-FSLP+ - (4.17)					R-FSLP+ - (4.19)					R-FSLP+ - (6.16)				
$\|C\|$	$\|CT\|$	$\|Q\|$	D	x^{Root}	x^{LB}	x^{UB}	$Gap(x^{UB})$	$t(s)$	x^{Root}	x^{LB}	x^{UB}	$Gap(x^{UB})$	$t(s)$	x^{Root}	x^{LB}	x^{UB}	$Gap(x^{UB})$	$t(s)$
60	10	2	*LD*	1530	1560	1710	8.77%	10800	1530	1562	1665	6.19%	10800	1530	1560	1710	8.77%	10800
60	10	2	*S*	1020	**1020**	1020	0.00%	207	1020	**1020**	**1020**	0.00%	252	1020	**1020**	**1020**	0.00%	26
60	10	2	*U*	1652	**1680**	1825	7.95%	10800	1652	**1680**	1900	11.58%	10800	1652	**1680**	1830	8.20%	10800
60	25	2	*LD*	2060	2141	**2250**	4.86%	10800	2060	2180	2265	3.75%	10800	2060	**2250**	**2250**	0.00%	5468
60	25	2	*S*	1360	1360	**1395**	2.51%	10800	1360	1360	1420	4.23%	10800	1360	**1361**	**1395**	2.45%	10800
60	25	2	*U*	1490	**1490**	**1490**	0.00%	228	1490	**1490**	**1490**	0.00%	251	1490	**1490**	**1490**	0.00%	297
			Average				4.01%	7272				4.29%	7284				3.24%	6365
240	20	2	*LD*	7880	**7880**	10920	27.84%	10800	7880	**7880**	11065	28.78%	10800	7880	**7880**	**10760**	26.77%	10800
240	20	2	*S*	4440	**4440**	5040	11.90%	10800	4440	**4440**	5040	11.90%	10800	4440	**4440**	**4950**	10.30%	10800
240	20	2	*U*	6720	**6720**	9380	28.36%	10800	6720	**6720**	9380	28.36%	10800	6720	**6720**	**8425**	20.24%	10800
240	60	2	*LD*	8260	**8260**	10285	19.69%	10800	8260	**8260**	10630	22.30%	10800	8260	8260	10415	20.69%	10800
240	60	2	*S*	5280	**5280**	**6535**	19.20%	10100†	5280	**5280**	6540	19.27%	10800	5280	**5280**	6660	20.72%	10800
240	60	2	*U*	7580	**7610**	**9435**	19.34%	10500†	7580	**7610**	10380	26.69%	10800	7580	**7610**	9790	22.27%	8000†
			Average				21.06%	10633				22.88%	10800				20.16%	10333
500	20	4	*LD*	14420	**14420**	**19820**	27.25%	10800	14420	**14420**	20820	30.74%	10800	14420	**14420**	22540	36.02%	10800
500	20	4	*S*	8250	**8250**	11775	29.94%	10800	8250	**8250**	11775	29.94%	10800	8250	**8250**	**11300**	26.99%	10800
500	20	4	*U*	14380	**14410**	19975	27.86%	10800	14380	14380	22180	35.17%	10800	14380	14380	20870	31.10%	10800
500	60	4	*LD*	14490	**14520**	18075	19.67%	10800	14490	**14520**	19065	23.84%	10800	14490	**14520**	**17850**	18.66%	10800
500	60	4	*S*	11840	**11840**	14035	15.64%	10800	11840	**11840**	16065	26.30%	10800	11840	**11840**	**14095**	16.00%	10800
500	60	4	*U*	15530	15530	**19460**	20.20%	10800	15530	15530	23130	32.86%	10800	15530	**15560**	20200	22.97%	10800
500	100	4	*LD*	15420	**15450**	18285	15.50%	10800	15420	**15450**	19960	22.60%	10800	15420	**15450**	**18000**	14.17%	10800
500	100	4	*S*	9490	**9490**	10575	10.26%	10800	9490	**9490**	11125	14.70%	10800	9490	**9490**	10790	12.05%	10800
500	100	4	*U*	16260	**16290**	20000	18.55%	10800	16260	**16290**	20100	18.96%	10800	16260	16260	**18650**	12.82%	10800
			Average				20.54%	10800				26.12%	10800				21.20%	10800
1000	20	4	*LD*	21970	**21990**	-	-	10800	21970	**21990**	-	-	10800	21970	**21990**	74560	70.51%	10800
1000	20	4	*S*	14570	**14570**	-	-	10800	14570	**14570**	-	-	10800	14570	**14570**	-	-	10800
1000	20	4	*U*	25940	**25980**	315260	91.76%	10800	25940	**25980**	**58680**	55.73%	10800	25950	**25980**	660910	96.07%	10800
1000	60	4	*LD*	27080	**27100**	-	-	10800	27080	**27100**	409460	93.38%	10800	27080	**27100**	**63465**	57.30%	10800
1000	60	4	*S*	17950	**17950**	56485	68.22%	10800	17950	**17950**	288695	93.78%	10800	17950	**17950**	-	-	10800
1000	60	4	*U*	40260	**40260**	405255	90.07%	10800	40260	**40260**	126105	68.07%	10800	40260	**40260**	**85390**	52.85%	10800
1000	100	4	*LD*	28750	**28770**	62720	54.13%	10800	28750	**28770**	**55870**	48.51%	10800	28750	**28770**	674000	95.73%	10800
1000	100	4	*S*	17840	**17840**	-	-	10800	17840	**17840**	349010	94.89%	10800	17840	**17840**	**30920**	42.30%	10800
1000	100	4	*U*	32210	**32210**	72745	55.72%	10800	32210	**32210**	286495	88.76%	10800	32210	**32210**	250080	87.12%	10800
			Average				71.98%	10800				77.59%	10800				71.70%	10800

6.5.2 Results for the hybrid heuristic

When testing the hybrid heuristic, the following values are used

$$|\Pi| = 30 \qquad \rho = 25\% \qquad n_{GRASP} = 1$$

As for the stopping criteria we use $n = 100,000$. The implementation of the hybrid heuristic uses the same implementation of the GRASP heuristic as the one described in Chapter 4 and the parameter values are reused as well. To account for the randomness, the hybrid heuristic is executed 10 times for each instance. Table 6.4 shows the results for the hybrid heuristic, and compares with the results from the GRASP method as presented in Chapter 4.[1] The first four columns in Table 6.4 reports the instance characteristics similar to Table 6.2-6.3. The next three columns show the best known lower bound, and best known upper bound from Table 6.2 and Table 6.3, and the gap between these. For the two heuristics Table 6.4 reports the value of the best-found solution (x^b), the average value of the solutions ($\bar{x}$), and the best and average gap ($Gap(x^b)$, $Gap(\bar{x})$) with respect to best known lower bound. The columns $\bar{t}$ (s) is the average time spent in seconds.

Table 6.4: Heuristic results: Comparison between the GRASP heuristic and the hybrid heuristic

				Best Known			GRASP					Hybrid				
$\|C\|$	$\|CT\|$	$\|Q\|$	D	x^{LB}	x^{UB}	$Gap(x^{UB})$	x^b	$\bar{x}$	$Gap(x^b)$	$Gap(\bar{x})$	$\bar{t}$ (s)	x^b	$\bar{x}$	$Gap(x^b)$	$Gap(\bar{x})$	$\bar{t}$ (s)
60	10	2	*LD*	1563	1665	6.10%	1805	1814.5	13.38%	13.83%	9.9	1795	1797	12.90%	13.00%	9.8
60	10	2	*S*	1020	1020	0.0%	1060	1067	3.77%	4.40%	7.5	1020	1032	0.00%	1.15%	8.2
60	10	2	*U*	1680	1770	5.08%	1895	1902.5	11.35%	11.69%	8.8	1845	1867	8.94%	10.01%	9.1
60	25	2	*LD*	2250	2250	0.0%	2435	2435	7.60%	7.60%	10.4	2435	2435	7.60%	7.60%	9.4
60	25	2	*S*	1361	1395	2.45%	1425	1436.5	4.50%	5.27%	10.1	1395	1413	2.45%	3.69%	9.3
60	25	2	*U*	1490	1490	0.0%	1545	1552.5	3.56%	4.02%	8.5	1510	1527.5	1.32%	2.45%	8.3
			Average			**2.27%**			**7.36%**	**7.80%**	**9.2**			**5.54%**	**6.32%**	**9.0**
240	20	2	*LD*	7880	10760	26.77%	9430	9448	16.44%	16.60%	80.5	9365	9391.5	15.86%	16.09%	66.7
240	20	2	*S*	4440	4950	10.30%	4790	4807.5	7.31%	7.64%	36.2	4720	4757.5	5.93%	6.67%	33.5
240	20	2	*U*	6720	8425	20.24%	8140	8333.5	17.44%	19.35%	36.8	7735	7955.5	13.12%	15.50%	34.8
240	60	2	*LD*	8260	10240	19.34%	10105	10123	18.26%	18.40%	131.1	10085	10105.5	18.10%	18.26%	70.3
240	60	2	*S*	5280	6535	19.20%	5660	5706.5	6.71%	7.47%	53.4	5520	5604	4.35%	5.78%	45.8
240	60	2	*U*	7610	9435	19.34%	9065	9140.5	16.05%	16.74%	52.8	8675	8889	12.28%	14.38%	41.2
			Average			**19.20%**			**13.70%**	**14.37%**	**65.1**			**11.61%**	**12.78%**	**48.7**
500	20	4	*LD*	14420	19820	27.25%	15585	15639	7.48%	7.79%	538.7	15400	15528	6.36%	7.13%	258.9
500	20	4	*S*	8250	11300	26.99%	9020	9129.5	8.54%	9.63%	75.0	8865	8925	6.94%	7.56%	72.7
500	20	4	*U*	14410	19120	24.63%	15585	15594.5	7.54%	7.60%	381.9	15400	15482	6.43%	6.92%	201.5
500	60	4	*LD*	14520	17850	18.66%	16130	16225.5	9.98%	10.51%	648.3	15950	16022.5	8.97%	9.38%	289.3
500	60	4	*S*	11840	14035	15.64%	13005	13080	8.96%	9.48%	72.2	12510	12718	5.36%	6.89%	69.4
500	60	4	*U*	15560	19460	20.04%	17125	17173	9.14%	9.39%	287.1	17075	17111	8.87%	9.06%	176.1
500	100	4	*LD*	15450	18000	14.17%	17475	17562	11.59%	12.03%	627.7	17185	17324.5	10.10%	10.82%	286.3
500	100	4	*S*	9490	10575	10.26%	10425	10547.5	8.97%	10.02%	73.6	10190	10415	6.87%	8.87%	71.2
500	100	4	*U*	16290	18650	12.65%	18495	18544	11.92%	12.15%	282.8	18290	18379	10.93%	11.37%	181.2
			Average			**18.92%**			**9.35%**	**9.84%**	**331.9**			**7.87%**	**8.67%**	**178.5**
1000	20	4	*LD*	21990	48185	54.36%	24225	24277	9.23%	9.42%	1984.0	24130	24201.5	8.87%	9.14%	818.6
1000	20	4	*S*	14570	-	-	16230	16414	10.23%	11.23%	292.2	15945	16152.5	8.62%	9.79%	237.8
1000	20	4	*U*	25980	58680	55.73%	28295	28354.5	8.18%	8.37%	1730.9	28235	28303.5	7.99%	8.21%	783.5
1000	60	4	*LD*	27100	63465	57.30%	30000	30074	9.67%	9.89%	2821.9	29810	29882.5	9.09%	9.31%	1159.2
1000	60	4	*S*	17950	50210	64.25%	20305	20599	11.60%	12.85%	174.8	19590	20244	8.37%	11.31%	171.7
1000	60	4	*U*	40260	85390	52.85%	44055	44095.5	8.61%	8.70%	1209.3	43855	43991.5	8.20%	8.48%	618.3
1000	100	4	*LD*	28770	55870	48.51%	32330	32372	11.01%	11.13%	3129.7	32040	32206.5	10.21%	10.67%	1220.7
1000	100	4	*S*	17840	30920	42.30%	20150	20347	11.46%	12.32%	177.2	19830	20057.5	10.04%	11.05%	172.9
1000	100	4	*U*	32210	66580	51.62%	36160	36228	10.92%	11.09%	2194.8	35940	36014.5	10.38%	10.56%	804.8
			Average			**53.37%**			**10.10%**	**10.56%**	**1523.9**			**9.08%**	**9.84%**	**665.3**
Average						**25.04%**			**10.05%**	**10.55%**	**571.6**			**8.51%**	**9.37%**	**264.7**

[1] Compared with Chapter 4, some of the gaps for the GRASP method are lower here. This is due to the improvement of the lower bounds.

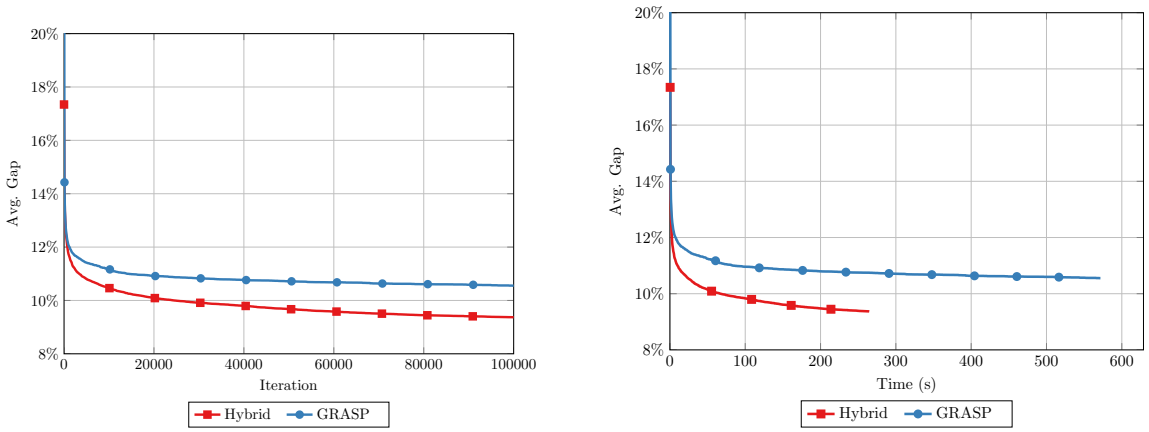
The results in Table 6.4 shows that the hybrid heuristic finds solutions with an average gap of 9.37%, in approx. 5 min on average. When evaluating the heuristic based on the gap, we need to account for the quality of the lower bound. For the smallest instances ($|C| = 60$) the best-known gap is only 2%, and thus we have reason to believe the lower bounds for those instances are better than for the rest of the instances. Now, focusing on the hybrid heuristic for the smallest instances we see that the gap here is lower compared to the rest of the instances. Leaving us to believe that there is as much further research to be done on improving the lower bounds as there is to improve the heuristic.

Comparing the GRASP and the hybrid heuristic, we see that the hybrid heuristic finds better solutions using approximately half of the time. The expected number of GRASP iterations for the hybrid heuristic is

$$|\Pi| + 2n\rho n_{GRASP} = 50030$$

The GRASP iterations are the most computationally expensive part of the method, and therefore it is expected that the hybrid heuristic is faster, as fewer GRASP iterations are performed compared to the GRASP standalone heuristic presented in Chapter 4.

Figure 6.4 shows how the two heuristics converge towards the final solution. The data is grounded on ten runs of each instance. Figure 6.4a shows how the two heuristics converge per iteration, and Figure 6.4b is the convergence over time. The graphs show what we also see in Table 6.4; i.e. the hybrid heuristic finds better solutions, using only half of the time.



(a) Iteration based convergence *(b) Time based convergence*

Figure 6.4: Convergence of the GRASP heuristic and the hybrid heuristic.

6.5.3 Column generation results

The column generation procedure as described in Section 6.4 solves an LP-relaxation and computes a lower bound. Therefore the results are compared with the linear relaxation of the models FSLP+ (LP-FSLP+) and R-FSLP (LP-R-FSLP). Table 6.5 shows the result. For the three test models, x is the value of the linear relaxation, and t is the time in seconds used to solve the relaxation. Along with these the column generation method also reports the number of iterations (Ite), the number of columns added to the final master problem ($|\Omega_{gen}|$) and the time spent generating variables in the pricing problem t_{price}.

Table 6.5: Results for the column generation procedure.

				LP-FSLP+		LP-R-FSLP		Column Generation				
$\|C\|$	$\|CT\|$	$\|Q\|$	D	x	t(s)	x	t(s)	Ite	$\|\Omega_{gen}\|$	x	t_{price}(s)	t(s)
60	10	2	*LD*	1500	0.38	1530	0.07	195	394	1500	29.59	31.98
60	10	2	*S*	1020	0.32	1020	0.05	196	394	1020	21.67	23.98
60	10	2	*U*	1640	0.33	1645	0.05	251	481	1640	41.04	44.81
60	25	2	*LD*	2030	0.14	2060	0.02	135	269	2030	12.15	13.58
60	25	2	*S*	1360	0.17	1360	0.02	169	333	1360	8.57	10.00
60	25	2	*U*	1490	0.25	1490	0.04	158	316	1490	11.42	12.92

Table 6.5 shows that the column generation procedure finds the same lower bound as the LP-relaxation of the model FSLP+ from Chapter 4. For half of the instances, the LP-relaxation of R-FSLP gives a stronger lower bound. Looking at the times for the column generation method, it can be seen that most of the time is spent in the pricing problem. Comparing the times for the three methods, we can see that the column generation method is much slower than the two others.

6.6 Conclusion

In this chapter we have proposed a set of new solution approaches for the Flexible Ship Loading Problem. The studied problem aims to integrate terminal-oriented stowage planning with the routing and scheduling of transfer vehicles. First, we have formulated a mathematical model for the problem, which outperforms the state-of-the-art. The revised model both finds better solutions and improves the previously best known lower bounds. Secondly, a novel hybrid heuristic has been proposed. The hybrid heuristic is a hybridization between the GRASP heuristic presented in Chapter 4, and a genetic algorithm. The heuristic improves the state-of-art and does so using considerably less computational effort. In the results section, it is argued that the lower bounds found by the mathematical model are of poor quality for the medium to large instances. To lay the foundation for improving the lower bounds, a column-generation algorithm has been described. However, the column-generation algorithm does not improve the lower bounds. To improve the lower bounds, we plan to devise different cuts to add to the existing decomposed model, or by decomposing the problem differently. Another way to decompose the problem could be to generate *crane-plans* instead of *vehicle-plans*, as is done currently. Doing so the time aspect can be handled in the pricing problem, making the master problem easier. The pricing problem will, however, be more complex, but it should also correspond to an improved lower bound.

Additional ideas for further research is to go beyond some of the underlying assumptions of this problem, e.g. include the load sequencing as part of the decision, or allowing a vehicle to transport containers to be loaded by different QCs, i.e. allowing for vehicle pooling. The decomposed model presented here can handle vehicle pooling naturally. The master problem will largely remain the same, and only the subproblem will need to change to allow for vehicle pooling.